TUGAS AKHIR - KI141502

STEGANOGRAFI TEKS PADA AKSARA SUNDA DENGAN PENDEKATAN LINE SHIFT CODING

Muhsin Bayu Aji Fadhillah
NRP 05111440000071

Dosen Pembimbing I
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

Dosen Pembimbing II
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – KI141502

STEGANOGRAFI TEKS PADA AKSARA SUNDA DENGAN PENDEKATAN LINE SHIFT CODING

Muhsin Bayu Aji Fadhillah
NRP 05111440000071

Dosen Pembimbing I
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

Dosen Pembimbing II
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – KI141502

TEXT STEGANOGRAPHY ON SUNDANESE SCRIPT USING LINE SHIFT CODING APPROACH

Muhsin Bayu Aji Fadhillah
NRP 05111440000071

Supervisor I
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

Supervisor II
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
Faculty of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

Steganografi Teks pada Aksara Sunda dengan Pendekatan Line Shift Coding

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Berbasis Jaringan
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :
Muhsin Bayu Aji Fadhillah
NRP : 05111440000071

Disetujui oleh Dosen Pembimbing Tugas Akhir :

HENNING TITI CIPTAYANI, S.Kom., M.Kom. ..... (pembimbing 1)
NIP: 19840708201012200

Dr. Eng. RADITYO ANGGORO, S.Kom., M.Sc. ..... (pembimbing 2)
NIP: 198410162008121002

SURABAYA
JUNI 2018

[Halaman ini sengaja dikosongkan]

STEGANOGRAFI TEKS PADA AKSARA SUNDA DENGAN PENDEKATAN LINE SHIFT CODING

Nama Mahasiswa : Muhsin Bayu Aji Fadhillah
NRP : 05111440000071
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Dosen Pembimbing 2 : Dr. Eng. Radityo Anggoro, S.Kom, M.Sc.

ABSTRAK

Keamanan informasi adalah faktor penting dalam teknologi informasi dan komunikasi. Kerahasiaan suatu informasi perlu dijaga agar tidak terjadi pencurian informasi. Sebagai bentuk pengamanan informasi, steganografi bisa digunakan sebagai bentuk penyembunyian informasi tanpa diketahui orang lain.

Steganografi memungkinkan informasi rahasia disisipkan melalui media teks, video, gambar dan audio. Steganografi teks dapat dilakukan dengan mengubah format teks ataupun mengubah karakter teks tertentu tanpa mengubah makna dari teks tersebut. Aksara Sunda dapat digunakan sebagai media alternatif untuk menjadi *cover text* dalam teknik steganografi teks. Aksara Sunda sendiri sudah terdapat dalam versi resmi *font* Unicode.

Teknik steganografi teks memiliki kerentanan untuk mengalami kehilangan informasi akibat dari sebuah serangan atau kerusakan seperti proses pencetakan dengan printer dan pencetakan ulang dengan mesin *fotocopy*.

Metode yang ditawarkan pada tugas akhir ini menggunakan *line shift coding* untuk mengurangi akibat kerusakan yang ditimbulkan dari proses pencetakan dan pemindaian sehingga menghasilkan *stego text* yang memiliki tingkat *robustness* yang baik. *Line shift coding* menyisipkan bit biner pesan rahasia dengan menggeser posisi baris secara vertical. Pilihan pergeseran baris pada tugas akhir ini terbagi menjadi opsi geser baris genap yang menggeser baris genap, opsi geser baris 1-5 yang menggeser baris

diantara baris pertama dan kelima serta kelipatannya dan opsi geser baris 1-4-7 yang menggeser baris diantara baris pertama, keempat dan ketujuh serta kelipatannya.

Hasil uji coba metode menunjukkan bahwa steganografi teks pada Aksara Sunda berhasil dilakukan dengan mengganti blok Unicode karakter pada baris yang bergeser. *Capacity ratio* rata-rata yang dihasilkan adalah 6,06 bit/kiloByte. *Stego text* yang dihasilkan juga mampu bertahan dari pencetakan dengan printer dan proses *fotocopy* hingga dua kali pencetakan ulang untuk mempertahankan pesan rahasia yang dikandungnya.

Kata kunci: **Steganografi**, **Steganografi Teks, Aksara Sunda,** ***Line shift coding, Robustness*****.**

TEXT STEGANOGRAPHY ON SUNDANESE SCRIPT USING LINE SHIFT CODING

Student Name	: Muhsin Bayu Aji Fadhillah
Student ID	: 05111440000071
Major	: Informatics Department FTIf-ITS
Advisor 1	: Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Advisor 2	: Dr. Eng. Radityo Anggoro, S.Kom, M.Sc.

ABSTRACT

Information security is an important factor in information and communication technology. Confidentiality of an information needs to be maintained in order to avoid information theft. As a form of information security, steganography can be used as a form of concealment of information unnoticed by others.

Steganography allows confidential information to be inserted through text, video, image and audio media. Text steganography can be done by changing the text format or change the character of a particular text without changing the meaning of the text. Sundanese script can be used as an alternative media to be a cover text in text steganography techniques. Sundanese script itself is already contained in the official version of Unicode font.

Text steganography techniques have the vulnerability to experience the loss of information resulting from an attack or malfunction such as the printing process with the printer and reprinting it with a photocopier.

The method offered in this final project using line shift coding to reduce the damage caused by the printing and scanning process so as to produce stego text that has good robustness level. Line shift coding inserts binary bits of secret messages by sliding the vertical row position. The line shifting option in this final project is divided into even-numbered drag options that shift the even row, a row of 1-5 lines sliding option that shifts the rows between the first and fifth rows and their multiples and the 1-4-7

sliding line option that shifts the rows between the first rows, fourth and seventh and multiples of it.

The experimental results of the method show that the text steganography on Sundanese script is done by replacing the Unicode blocks of characters on the shifted row. The average capacity ratio generated is 6,06 bits / kiloByte. The resulting stego text is also able to withstand printing process with the printer and copying process up to two reprints to retain the secret messages it contains.

Keywords**: **Steganography**, **Text Steganography, Sundanese script, Line shift coding, Robustness.

KATA PENGANTAR

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

Puji syukur kehadirat Allah SWT karena atas limpahan karunia dan rahmatnya tugas akhir dengan judul **"Steganografi Teks pada Aksara Sunda dengan Pendekatan Line Shift Coding"** ini dapat terselesaikan.

Terimakasih yang sebesar – besarnya saya ucapkan kepada berbagai pihak yang telah mendukung dan membantu saya dalam pengerjaan tugas akhir ini, antara lain :

1. Orang Tua dan keluarga saya yang selalu memberi doa, dukungan dan motivasi untuk menyelesaikan tugas akhir ini.
2. Ibu Henning Titi Ciptaningtyas, S.Kom., M.Kom., selaku dosen pembimbing 1.
3. Bapak Dr. Eng. Radityo Anggoro, S.Kom, M.Sc., selaku dosen pembimbing 2.
4. Paman Sentot Susanto yang telah banyak membantu penulis hingga selesainya pengerjaan tugas akhir ini.
5. Kakek dan nenek penulis.
6. Saudara Cahya Setya Adhi, selaku teman yang senantiasa menjadi mentor dalam masa pembuatan tugas akhir ini.
7. Mas Mohamad Rijal Abdul Halim, Mas Faizal Anugrah Bhaswara dan Mas Zikrul Ihsan yang telah memberi banyak nasihat, motivasi, petunjuk dan dokumentasi.
8. Saudara Ahmad Samsul Choiruddin, yang sudah banyak membantu dalam pengujian tugas akhir ini.
9. Saudara Kunto, Chatra, Erza dan seluruh anggota Koloni Sarean.
10. Seluruh anggota Ashabul Roudhlotul Jannah.

11. Seluruh keluarga Muda-Mudi Masjid Luhur Al-Ikhlas Semampir.
12. Seluruh keluarga Mahasiswa Teknik Informatika ITS angkatan 2014.
13. Seluruh rekan Tim Futsal Informatika ITS.
14. Dan seluruh pihak yang telah membantu dan mendukung.

Semoga tugas akhir ini dapat dimanfaatkan dengan sebagai mestinya, dan bermanfaat bahkan setelah tugas akhir ini selesai untuk Almamater, masyarakat dan bangsa.

Surabaya, 1 Juni 2018
Penulis

Muhsin Bayu Aji Fadhillah
05111440000071

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1. Latar Belakang

Keamanan informasi adalah faktor penting dalam teknologi informasi dan komunikasi. Kerahasiaan suatu informasi perlu dijaga agar tidak terjadi pencurian informasi. Sebagai bentuk pengamanan informasi, terdapat beberapa cara yang bisa digunakan untuk menyembunyikan informasi rahasia. Salah satu cara untuk menyembunyikan informasi adalah steganografi. Kata steganografi sendiri berasal dari Bahasa Yunani "stegos" yang berarti sampul dan "grafia" yang berarti menulis [1]. Steganografi memungkinkan informasi rahasia disisipkan melalui berbagai macam bentuk media seperti teks, video, gambar dan audio sehingga orang awam tidak mengetahui bahwa ada informasi yang disisipkan tanpa menggunakan metode khusus untuk membuka informasi.

Steganografi teks termasuk jenis yang paling sulit karena tingkat redundasi informasi yang rendah dibandingkan media audio dan gambar [2]. Steganografi teks dapat dilakukan dengan mengubah format teks ataupun mengubah karakter teks tertentu tanpa mengubah makna dari teks tersebut [3]. Perubahan karakter teks yang terjadi harus dilakukan tanpa membuat pembaca awam curiga atau menyadari ada sesuatu yang berubah.

Ada beberapa alternatif yang dapat digunakan sebagai *cover text*. Vidhya dan Vargese menggunakan Malayalam, salah satu Bahasa lokal di India menjadi *cover text* sebagai metode alternatif untuk steganografi teks [2]. Alla dan Prasad menggunakan Bahasa lokal lain di India, Telugu, untuk metode steganografi teks dengan memanfaatkan kesesuaian Ottulu dan tanda baca [4]. Salah satu

bahasa daerah di Indonesia yang bahkan sudah dibuat versi *font* unicodenya, Aksara Sunda, dapat dijadikan sebagai metode alternatif untuk *cover text* steganografi teks. Aksara Sunda sendiri bersifat silabik, yaitu satu huruf mewakili satu suku kata. Rarangkén adalah vokalisasi atau diakritik yang digunakan dalam Aksara Sunda untuk mengubah bunyi vokal. Rarangkén dapat ditulis di atas, di bawah, ataupun sejajar dengan huruf dari Aksara Sunda. Jumlah rarangkén dalam aksara sunda sendiri mencapai 13 buah.

Beberapa parameter dapat dijadikan acuan untuk menguji dan menilai hasil teknik Steganografi. Parameter yang dapat dijadikan acuan antara lain *Fidelity*, *Recovery*, *Robustness*, dan *Security* [5]. *Fidelity* adalah aspek dimana *stego text* hasil steganografi dan *cover text* terlihat tidak berbeda secara kasat mata. Tingkat fidelity dapat dilakukan dengan pengujian MOS (*Mean Opinion Score*) maupun dengan membandingkan ukuran file dari *stego text* dan *cover text* [5]. *Recovery* adalah aspek dimana informasi rahasia yang sudah disisipkan dapat diekstraksi kembali untuk mengetahui informasi rahasia tersebut [5]. *Robustness* adalah aspek untuk mengetahui tingkat ketahanan informasi rahasia yang disisipkan akibat serangan, ganggua, ataupun manipulasi [5]. Tingkat *robustness* dapat diuji dengan melakuan *scaling*¸ *cropping*, ataupun *fotocopy* pada dokumen *stego text*. Paramater *security* dapat dilihat dari tingkat ketahanan steganografi terhadap serangan steganalisis [6].

Teknik steganografi teks memiliki kerentanan untuk mengalami kehilangan informasi akibat dari sebuah serangan atau kerusakan. Proses pencetakan dengan *printer* dan pemindaian dengan *scanner* adalah contoh proses yang dapat menghilangkan pesan yang telah disisipkan pada dokumen teks. Ada beberapa metode yang dapat dipakai untuk steganografi teks, seperti *Line Shift Coding*, *Word Shift Coding*, dan *Feature Coding* [6]. *Line Shift Coding* dapat digunakan sebagai metode alternatif untuk mengurangi akibat kerusakan yang ditimbulkan dari proses pencetakan dan pemindaian sehingga menghasilkan tingkat

robustness yang baik [7]. Metode *Line Shift Coding* dapat dilakukan dengan cara menggeser baris dari *cover text* secara vertikal untuk menyisipkan informasi rahasia.

Dalam tugas akhir yang penulis ajukan ini akan digunakan pendekatan *Line Shift Coding* untuk metode penyisipan teks yang membuat teks hasil penyisipan mampu bertahan dari beberapa distorsi hasil pencetakan dan pemindaian serta penggunaan Aksara Sunda sebagai media cover text dari steganografi teks.

1.2. Perumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana tingkat ketahanan *stego text* yang dihasilkan dari proses steganografi terhadap serangan berupa pencetakan dan pemindaian?
2. Bagaimana tingkat kemiripan *stego text* yang dihasilkan dengan *cover text* ?
3. Bagaimana tingkat keutuhan informasi rahasia yang disisipkan ketika diekstraksi kembali?

1.3. Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan antara lain:

1. Aksara Sunda yang dipakai adalah Aksara Sunda baku.
2. *Font* Aksara Sunda yang dipakai adalah *font* Sundanese Unicode 2013.
3. *Cover text* dimasukkan secara manual, tidak melalui proses *generate* otomatis.
4. Pencetakan melalui *printer* dan proses *fotocopy* sebagai bentuk serangan pada dokumen *stegotext*.

1.4. Tujuan Penelitian

1. Menghasilkan *stego text* hasil proses steganografi teks yang tahan terhadap serangan pencetakan dan pemindaian.

2. Menghasilkan *stego text* hasil proses steganografi teks yang mirip dengan *cover text* sehingga keamanan informasi yang disisipkan lebih terjamin.
3. Menghasilkan sistem steganografi yang dapat menyisipkan informasi kemudian mendapatkan kembali informasi dengan utuh.

1.5. Manfaat Penelitian

Manfaat dari tugas akhir ini adalah menghasilkan sistem transmisi informasi melalui steganografi teks dengan Aksara Sunda sebagai *cover text* dan pendekatan *Line Shift Coding* yang menghasilkan *stego text* hasil steganografi yang tahan terhadap serangan berupa pencetakan dan pemindaian.

1.6. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

1. Studi Literatur
   Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan program yaitu mengenai Steganografi, Aksara Sunda, *cover text*, pendekatan line shift coding, dan tingkat *robustness*, *fidelity*, serta *recovery* dari hasil steganografi
2. Analisis dan Desain Perangkat Lunak
   Analisis kebutuhan dan perancangan sistem dilakukan untuk merumuskan solusi yang tepat dalam pembuatan sistem yang dapat dilakukan untuk mengimplementasikan rancangan. Perancangan berisi alur kerja sistem dan data yang digunakan serta desain antarmuka dari sistem yang dibangun
3. Implementasi Perangkat Lunak
   Steganografi teks pada aksara Sunda akan dibuat dengan bahasa pemrograman Java menggunakan perangkat pengembangan NetBeans IDE pada platform desktop.

Library yang digunakan pada sistem yaitu Apache POI dan Apache PDFBox.

4. Uji Coba dan Evaluasi
   Pengujian dilakukan dengan melakukan proses ekstraksi kepada dokumen hasil steganografi. Dilakukan juga beberapa perusakan informasi steganografi pada dokumen cetak dengan melakukan proses *fotocopy*. Dari hasil pengujian, akan dihitung bagaimana tingkat keberhasilan proses ekstraksi dan tingkat *robustness* dari teknik steganografi yang dilakukan.

1.7. Sistematika Laporan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan
Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Tinjauan Pustaka
Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

Bab III Analisis dan Perancangan Sistem
Bab ini membahas mengenai perancangan perangkat lunak. Perancangan perangkat lunak meliputi perancangan data, arsitektur, proses dan perancangan antarmuka pada sistem.

Bab IV Implementasi
Bab ini berisi implementasi dari perancangan perangkat lunak.

Bab V Pengujian dan Evaluasi

Bab ini membahas tahap-tahap uji coba. Kemudian hasil uji coba dievaluasi untuk kinerja dari aplikasi yang dibangun.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

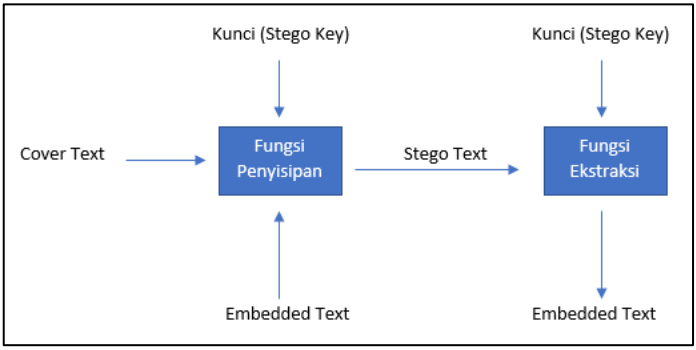
BAB II
TINJAUAN PUSTAKA

Bab ini menjelaskan tentang dasar-dasar teori yang menjadi dasar pembuatan tugas akhir dengan tujuan untuk memberikan gambaran-gambaran umum terhadap penelitian yang sedang dikerjakan.

2.1. Steganografi

Steganografi adalah seni dan ilmu menulis pesan tersembunyi atau menyembunyikan pesan dengan suatu cara. Kata steganografi sendiri berasal dari Bahasa Yunani "stegos" yang berarti sampul dan "grafia" yang berarti menulis [1].

Tidak seperti enkripsi yang pesannya disembunyikan, tetapi orang awam tau bahwa ada pesan yang disembunyikan, dalam teknik steganografi, bukan hanya informasi yang disisipkan yang menjadi kerahasiaan, adanya informasi rahasia sendiri sudah menjadi kerahasiaan. Kerahasiaan adanya informasi rahasia yang ada pada steganografi mengharuskan *stego text* hasil steganografi yang baik tidak diketahui orang lain selain si pembuat dan si penerima pesan.



Gambar 2.1 Gambaran proses dalam steganografi

Gambaran proses dalam sistem steganografi seperti ditunjukkan pada Gambar 2.1. *Cover text* merupakan media sebagai wadah yang digunakan dalam penyisipan informasi rahasia. *Embedded text* merupakan informasi rahasia yang disisipkan. *Embedded text* bisa juga disebut *secret message*. *Stego text* adalah hasil keluaran dari proses penyisipan informasi, *stego text* akan terlihat sama dengan *cover text* dan akan semakin baik bila perbedaan pada *stego text* dan *cover text* sulit terlihat secara kasat mata. *Stego key* merupakan istilah yang digunakan untuk menyebut kunci rahasia yang dipakai untuk menyembunyikan informasi rahasia maupun untuk mengekstraksi informasi rahasia dari *cover text* [8].

Pada tugas akhir ini teknik steganografi dipilih sebagai cara menyembunyikan informasi. *Embedded text* yang digunakan berupa karakter latin yang disisipkan pada *cover text* berupa media teks berbentuk docx dengan metode penyisipan *line shift coding*. *Cover text* yang digunakan dan *stego text* yang dihasilkan merupakan berkas teks Aksara Sunda yang disimpan dalam bentuk PDF. *Stego key* pada tugas akhir ini menggunakan *font* Sundanese Unicode 2013 yang sudah dimodifikasi dan opsi geser pada baris kalimat.

2.2. Steganografi Teks

Steganografi teks adalah salah satu jenis steganografi yang dapat dilakukan dengan mengubah format teks atau karakteristik tertentu dari elemen tekstual. Tujuan yang diperlukan dalam perancangan sistem adalah membuat pengkodean yang dapat diartikan kembali dengan membuat perubahan seminimal mungkin sehingga perubahan yang terjadi tidak terhiraukan oleh pembaca.

Steganografi teks dapat diklasifikasikan secara luas menjadi tiga jenis, yaitu *Format Based Methods, Random and Statistical Generation* serta *Linguistik methods* [9].

Format Based Methods mengubah bentuk fisik format teks untuk menyembunyikan informasi. Metode juga memiliki kekurangan. Jika *stego text* dibuka dengan pengolah kata, salah eja

dan spasi ekstra putih akan terdeteksi. Ukuran dari *font* yang berubah juga dapat menimbulkan kecurigaan pada manusia pembaca. Selain itu, jika *cover text* tersedia, komparasi *cover text* dengan *stego text* membuat bagian-bagian teks yang dimanipulasi cukup terlihat [10].

Random and Statistical Generation akan menyembunyikan informasi secara acak dengan memerhatikan sifat statistik dari karakter yang muncul dalam teks [9].

Linguistik methods secara khusus mempertimbangkan sifat linguistik dari *stego text* yang dihasilkan dan *cover text* yang dimodifikasi, dan dalam banyak kasus, menggunakan struktur linguistik sebagai ruang untuk menyembunyikan informasi [10].

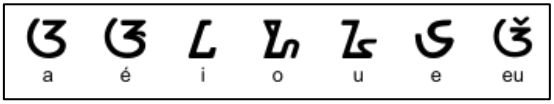
Terdapat beberapa macam pendekatan pada steganografi teks yang sudah ada, seperti *line shift coding*, *word shift coding*, dan *feature Coding* [3]. Pada tugas akhir ini, pendekatan yang digunakan adalah *line shift coding*.

2.3. Aksara Sunda

Berdasarkan Peraturan Daerah Tingkat 1 Jawa Barat, Aksara Sunda adalah sistem ortografi hasil kreasi masyarakat Jawa Barat yang meliputi aksara dan sistem pengaksaraan untuk menuliskan bahasa Sunda. Aksara Sunda berjumlah 32 buah yang terdiri atas 7 aksara swara ‘vokal mandiri’ (a, é, i, o, u, e, dan eu) dan 23 aksara ngalagena ‘konsonan’ (ka-ga-nga, ca-ja-nya, ta-da-na, pa-ba-ma, ya-ra-la, wasa-ha, fa-va-qa-xa-za) [11]. Aksara Sunda juga memiliki 13 buah rarangkén. Rarangkén dapat ditulis di atas, di bawah, ataupun sejajar dengan huruf dari Aksara Sunda. Saat ini, Aksara Sunda telah memiliki *font* resmi versi *Unicode*-nya. *Font* Aksara Sunda Unicode adalah *font* yang didasarkan dari Aksara Sunda Baku. Pada tugas akhir ini, Aksara Sunda digunakan sebagai *stego key* untuk menyembunyikan informasi.

2.3.1. Aksara Swara (Vokal)

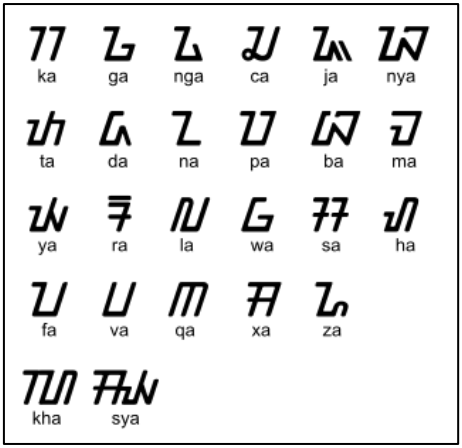
Aksara swara pada Aksara Sunda melambangkan bunyi vokal. Aksara swara dapat dipasangkan dengan Aksara ngalagena yang menjadi representasi karakter konsonan pada Aksara Sunda untuk membentuk sebuah kata. Aksara swara terdiri dari 7 buah aksara [11] seperti ditunjukkan pada Gambar 2.2.



Gambar 2.2 Aksara swara
(Sumber : kingsunda.com/aksara-sunda-kaganga-rarangken [12]**)**

2.3.2. Aksara Ngalagena

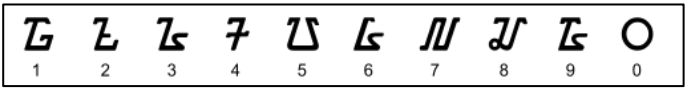
Aksara ngalagena melambangkan bunyi-bunyi fonem konsonan yang secara silabis mengandung bunyi vokal [11]. Bentuk aksara ngalagena seperti ditunjukkan pada Gambar 2.3.



Gambar 2.3 Aksara ngalagena
(Sumber : kingsunda.com/aksara-sunda-kaganga-rarangken [12]**)**

2.3.3. Aksara Angka

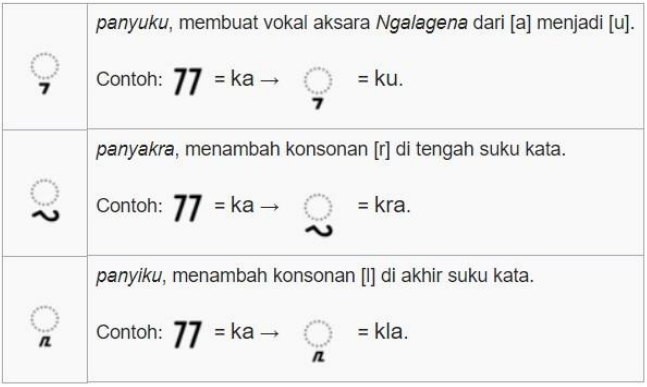
Aksara angka pada melambangkan angka-angka yang ada sama seperti pada huruf latin, terdapat 10 aksara yang melambangkan nilai “0” sampai “9” dengan penulisan dimulai dari kiri ke kanan. Bentuk beberapa lambang angka Sunda memiliki kemiripan dengan lambang aksara sehingga untuk penulisannya lambang angka harus diapit dengan garis vertikal yang lebih tinggi dari lambang angka. Bentuk aksara angka seperti ditunjukkan pada Gambar 2.4.



Gambar 2.4 Aksara angka
(Sumber : kingsunda.com/aksara-sunda-kaganga-rarangken [12]**)**

2.3.4. Rarangkén

Rarangkén adalah tanda vokalisasi pada Aksara Sunda [11]. Terdapat 14 buah rarangkén yang dibagi ke dalam tiga posisi penulisan.



Gambar 2.5 Rarangkén yang ditulis di bawah aksara dasar
(Sumber : kingsunda.com/aksara-sunda-kaganga-rarangken [12]**)**

Terdapat rarangkén yang penulisannya dapat ditempatkan di bawah aksara dasar. Ada tiga buah aksara yang penulisannya termasuk dalam posisi bawah seperti ditunjukkan pada Gambar 2.5.

Gambar 2.6 Rarangkén yang penulisannya di atas aksara dasar (Sumber : kingsunda.com/aksara-sunda-kaganga-rarangken [12]**)**

Selanjutnya, terdapat lima rarangkén yang penulisannya berada di posisi atas. Dua buah rarangkén juga dapat dituliskan bersamaan di atas sebuah aksara dasar. Penulisan rarangkén di posisi atas aksara dasar seperti ditunjukkan pada Gambar 2.6.

Terakhir, rarangkén dapat ditulis sejajar dengan aksara dasar. Terdapat enam buah rarangkén penulisannya bisa sejajar dengan aksara dasar seperti ditunjukkan pada Gambar 2.7.

	*panéléng*, membuat vokal aksara *Ngalagena* dari [a] menjadi [ɛ]. Contoh:  = ka →  = ké.
	*panolong*, membuat vokal aksara *Ngalagena* dari [a] menjadi [ɔ]. Contoh:  = ka →  = ko.
	*pamingkal*, menambah konsonan [j] di tengah suku kata. Contoh:  = ka →  = kya.
	*pangwisad*, menambah konsonan [h] di akhir suku kata. Contoh:  = ka →  = kah.
	*patén* atau *pamaéh*, meniadakan vokal pada suku kata. Contoh:  = ka → pamaeh = k.

Gambar 2.7 Rarangkén yang penulisannya sejajar aksara dasar (Sumber : kingsunda.com/aksara-sunda-kaganga-rarangken [12]**)**

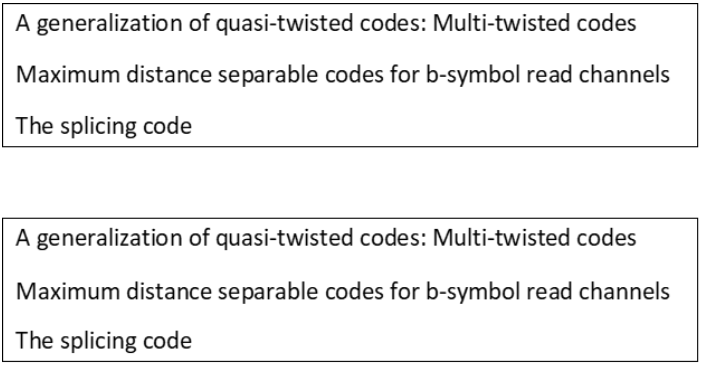
2.4. *Cover Text*

Cover text atau bisa disebut *cover media* adalah media yang dijadikan sebagai sebuah wadah untuk menyembunyikan pesan atau informasi yang disisipkan. *Cover text* adalah salah satu bagian dari properti steganografi. *Cover text* dalam steganografi bisa berbentuk file teks, gambar, audio, maupun video.

Properti lain dari steganografi selain *cover text* adalah *embedded message* atau pesan yang disembunyikan, *stego text* atau hasil keluaran dari proses penyisipan, dan *stego key* atau kunci rahasia yang digunakan dalam menyembunyikan informasi dan juga untuk mendapatkan kembali informasi dari media tempat informasi tersebut disembunyikan [8].

2.5. *Line Shift Coding*

Line shift coding adalah metode untuk mengubah dokumen dengan menggeser secara vertikal lokasi baris teks untuk mengkodekan dokumen. Pengkodean ini dapat diterapkan baik ke dalam format file atau ke bitmap gambar. *Codeword* yang sudah tertanam dapat diekstraksi dari format file atau bitmap. Dalam kasus tertentu proses *decoding* bisa dilakukan tanpa membutuhkan gambar asli, karena dari gambar yang asli diketahui memiliki jarak garis seragam antar garis yang berdekatan dalam paragraf [3].



Gambar 2.8 Paragraf atas sebelum line shift coding, gambar paragraf bawah setelah proses line shift coding

Contoh penerapan *line shift coding* seperti pada Gambar 2.8. Paragraf atas baris kedua berada dalam baris normal, tetapi di paragraf bawah, baris kedua sudah mengalami pergeseran posisi ke bawah untuk menyisipkan informasi. Baris ganjil menjadi *control groups*, dimana baris-baris ini tidak akan bergeser untuk mengurangi kemungkinan distorsi. Sementara, baris genap menjadi baris yang bergeser posisinya untuk disisipi informasi.

Line shift coding dianggap lebih mampu bertahan dari serangan, seperti distorsi akibat pemindaian dengan *scanner* ataupun pencetakan ulang dengan mesin *fotocopy*. Meskipun

begitu, pendekatan *line shift coding* memiliki kelemahan dari sisi kapasitas informasi yang bisa disisipkan karena harus menggeser posisi baris. Kapasitas informasi yang bisa disisipkan pada *line shift coding* lebih rendah dibanding dengan *word shift coding* yang menggeser posisi per kata pada *cover text* untuk penyisipan informasinya.

Andiniarti menggunakan baris genap sebagai *control groups* dalam penyisipan informasi dengan pendekatan *line shift coding* sehingga hanya baris-baris ganjil yang bisa digeser untuk penyisipan informasi [7].

Pada tugas akhir ini dibuatlah dua opsi geser baris lain untuk meningkatkan *capacity ratio*. Opsi pertama, baris pertama, kelima, dan selisih lima urutan baris dari baris pertama dan kelima akan digunakan sebagai *control groups*. Pada opsi kedua, baris pertama, keempat, dan ketujuh digunakan sebagai *control groups*. Selanjutnya tiap selisih 3 baris dibawah baris ketujuh akan menjadi *control groups* selanjutnya.

2.6. *Robustness*

Robustness adalah salah kriteria yang harus diperhatikan dalam proses steganografi. Kondisi yang baik adalah saat data yang ditambahkan atau disembunyikan pada citra file lain tidak rusak karena dilakukan beberapa operasi manipulasi pada file seperti perubahan kontras, penajaman, rotasi, pemampatan, pemotongan, maupun enkripsi. Steganografi yang sukses akan tetap melindungi file yang disembunyikan sehingga tidak berubah dan tetap valid walaupun telah dilakukan pemrosesan pada file penampung (*cover text*) [5].

Pada tugas akhir ini tingkat *robustness* dari steganografi akan diuji dengan proses *fotocopy stego text*, dimana hasil *fotocopy* akan diekstraksi kembali untuk menemukan *secret message* yang disembunyikan.

2.7. *Fidelity*

Fidelity adalah salah satu aspek penilaian baik tidaknya steganografi. Steganografi mengharuskan keluaran (*stego text*) terlihat sama dan tidak jauh berbeda dengan *cover text* yang menjadi masukan. Aspek inilah yang disebut *fidelity*, dimana *stego text* hasil steganografi dan *cover text* terlihat tidak berbeda secara kasat mata [5].

Pada tugas akhir ini, aspek *fidelity* pada steganografi diukur dengan melakukan pengujian MOS (*Mean Opinion Score*) dan dengan membandingkan ukuran file dari *stego text* dan *cover text*.

2.8. *Recovery*

Cover text yang sudah disisipi *secret message* akan menghasilkan *stego text*. Pada proses ekstraksi, *stego text* akan diproses kembali untuk mencari *secret message* yang disembunyikan. Aspek ekstraksi *stego text* inilah yang disebut *recovery*, dimana informasi rahasia yang sudah disisipkan dapat diekstraksi kembali dari *stego text* untuk mengetahui informasi rahasia tersebut [5].

Pada tugas akhir ini, tingkat *recovery* dari steganografi dinilai dari seberapa baiknya hasil ekstraksi *stego text* untuk mendapatkan kembali *secret message* secara utuh.

2.9. *Capacity Ratio*

Capacity ratio adalah ukuran kemampuan media yang dijadikan *cover text* dalam menampung *secret message*. Pada tugas akhir ini, *capacity ratio* dihitung untuk mengukur kemampuan teks Aksara Sunda dalam menyimpan bit informasi *secret message* dengan mengacu pada pendekatan *line shift coding*.

$$Capacity\ ratio = \frac{Text\ capacity}{Text\ ratio} \tag{2.1}$$

Penghitungan *capacity ratio* (bit/kilobyte) seperti ditunjukkan pada Formula 2.1, dimana *text capacity* (bit) merupakan kapasitas informasi yang disisipkan media *cover text*.

Text size (kilobyte) merupakan ukuran dari berkas yang disisipi informasi.

Pada tugas akhir ini, jumlah baris yang digeser dihitung sebagai *text capacity*, sementara ukuran berkas *cover text* yang berisi teks Aksara Sunda dihitung sebagai *text size*.

2.10. Java

Gambar 2.9 Logo Java
(Sumber : www.diylogodesigns.com [13]**)**

Bahasa pemrograman Java dibuat sebagai bahasa untuk tujuan umum, yang berbasis kelas, dan berorientasi objek. Java masih terkait dengan C dan C++ namun dibuat agak berbeda dengan beberapa aspek dari C dan C++ yang dihilangkan dan beberapa aspek dari bahasa lain yang ditambahkan [14].

Java sering digunakan karena kemudahan dalam proses pengembangan program. Java merupakan bahasa yang *multiplatform* sehingga bisa digunakan pada sistem operasi manapun. Selain itu, Java merupakan bahasa yang mendukung pemrograman berorientasi objek.

2.11. Netbeans

Gambar 2.10 Logo Netbeans *IDE*
(Sumber : www.pramudito.com [15]**)**

Netbeans adalah *IDE* (*Integrated Development Environment)* yang dikembangkan oleh Sun Microsystems. Netbeans adalah *IDE* resmi untuk Java 8 [16]. Dalam Netbeans, program atau aplikasi dikembangkan dari kumpulan komponen yang disebut modules. Pembuatan aplikasi Java dengan antarmuka grafis dapat dilakukan dengan Netbeans. Pada tugas akhir ini, Netbeans digunakan sebagai perangkar pengembang aplikasi.

2.12. Apache PDFBox

Gambar 2.11 Logo Apache PDFBox
(Sumber : www.sdtimes.com [17]**)**

Apache PDFBox adalah *library open source* pada Java yang bisa digunakan untuk bekerja dengan dokumen PDF. Fitur pada Apache PDFBox memungkinkan pembuatan dokumen PDF baru, manipulasi dokumen yang sudah ada, dan kemampuan untuk mengekstraksi konten yang ada pada dokumen. Apache PDFBox juga mencakup beberapa utilitas *command*-line [18].

2.13. Apache POI

Gambar 2.12 Logo Apache POI
(Sumber : www.waltercedric.com [19]**)**

Apache POI dalam *open source library* pada Java yang dapat digunakan untuk menanganani berkas atau dokumen yang dipakai pada Microsoft Office. *Library* Apache POI dapat digunakan untuk menulis dan membaca berkas Microsoft Office seperti word, excel, dan power point. Apache POI juga mendukung berkas pada Open Office XML [20].

2.14. Font Forge

Gambar 2.13 Logo FontForge
(Sumber : www.github.com/fontforge/fontforge [21]**)**

FontForge adalah perangkat lunak yang bisa digunakan untuk membuat *font* baru, mendesain, maupun mengedit *font* yang sudah ada. Penggunaan FontForge mampu mendukung banyak macam format *font*, seperti TrueType Collection (TTC), OpenType (OTF), TrueType (TTF), Web Open Font Format (WOFF), XML formats, TrueType XML (TTX) hingga Adobe Feature File (fea). FontForge sendiri merupakan *font* editor bersifat *open source* atau berlisensi gratis [22].

[Halaman ini sengaja dikosongkan]

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini menjelaskan tentang analisis dan perancangan sistem. Steganografi teks pada Aksara Sunda dengan pendekatan *line shift coding* adalah sistem yang dibangun pada tugas akhir ini. Sistem yang akan dibangun pada tugas akhir ini terdiri dari dua modul, yaitu modul *embedding* dan modul ekstraksi.

3.1. Data

Sistem yang akan dibangun memiliki dua modul dan setiap modul memiliki data masukan yang berbeda. Data yang menjadi masukan pada tiap modul akan diolah dengan metode yang digunakan untuk menghasilkan data keluaran.

3.1.1. Data Modul *Embedding*

Data masukan untuk modul *embedding* terdiri dari :

1. *Secret message*

   Secret message merupakan data masukan berupa informasi atau pesan yang akan disisipkan ke dalam cover text. *Secret message* ini berupa huruf latin.
2. *Cover text*

   Cover text merupakan berkas teks yang dipakai untuk menyisipkan dokumen. Teks yang dipakai dalam tugas akhir ini berupa teks Aksara Sunda dalam bentuk dokumen docx.
3. Opsi Geser

   Opsi geser merupakan pilihan untuk pergeseran posisi baris. Terdapat tiga opsi geser, yaitu opsi geser baris genap, opsi geser baris 1-5, dan opsi geser baris 1-4-7.

Data keluaran dari modul embedding adalah *stego text*. *Stego text* adalah teks yang telah disisipi *secret message*. *Stego*

text hasil keluaran disimpan dalam bentuk PDF yang berisi teks Aksara Sunda.

3.1.2. Data Modul Ekstraksi

Data masukan untuk modul ekstraksi terdiri dari :

1. *Stego text*

   Stego text merupakan teks yang telah disisipi *secret message*. Pada modul ekstraksi *stego text* akan diproses untuk mengetahui pesan rahasia yang terkandung di dalamnya.
2. Opsi Geser

   Opsi geser merupakan pilihan untuk pergeseran posisi baris. Opsi geser digunakan untuk menyesuaikan pencarian *secret message* yang tepat pada *stego text*.

Data keluaran dari modul ekstraksi adalah *secret message* yang didapat dari *stego text*. *Secret message* adalah pesan rahasia yang disisipkan pada teks *stego text*. *Secret message* akan ditampilkan setelah *stego text* berhasil diproses dalam modul ekstraksi.

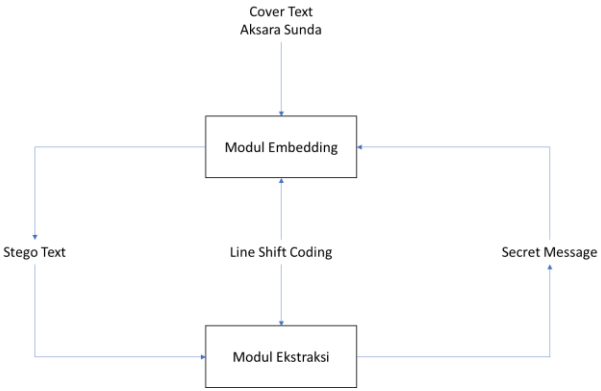
3.1.3. Data Font Modifikasi

Data *font* modifikasi adalah *font* Sundanese Unicode 2013 yang telah dimodifikasi dan digunakan pada tugas akhir ini. *Font* Sundanese Unicode 2013 dimodifikasi untuk menangani pergeseran baris yang dilakukan pada modul *embedding*. Hasil modifikasi *font* Sundanese Unicode 2013 juga memuat rarangkén sejajar yang diatur posisinya karena penulisan rarangkén sejajar *font* Sundanese Unicode 2013 standar untuk dokumen PDF pada lingkungan Java tidak didukung penuh oleh *library* Apache PDFBox. FontForge adalah perangkat lunak yang digunakan dalam proses modifikasi *font* Sundanese Unicode 2013.

Instalasi *Font* Sundanese Unicode 2013 modifikasi terlebih dahulu dilakukan agar sistem operasi dan sistem steganografi yang dibuat dapat mendeteksi *font*. Modifikasi dari *font* Sundanese Unicode 2013 akan digunakan baik pada modul *embedding* maupun modul ekstraksi pada sistem steganografi.

3.2. Deskripsi Umum Sistem

Pada tugas akhir ini, sistem steganografi pada Aksara Sunda akan memiliki dua modul, yaitu modul *embedding* dan modul ekstraksi. Sistem steganografi pada tugas akhir ini menggunakan pendekatan *line shift coding* melalui pergesaran posisi baris kalimat pada teks untuk metode penyisipan data. Skema umum sistem seperti ditunjukkan pada Gambar 3.1.



Gambar 3.1 Skema umum sistem

Modul *embedding* pada sistem ini akan menampilkan antarmuka untuk pengisian data masukan berupa *secret message* , opsi geser dan *cover text*. Tampilan antarmuka pada modul *embedding* juga akan menampilkan kode biner hasil translasi dari *secret message* yang dimasukkan. Modul *embedding* akan mengolah *secret message* untuk disisipkan pada *cover text*

sehingga keluarannya adalah *stego text* yang berisi informasi rahasia. Keluaran *stego text* hasil modul *embedding* adalah teks Aksara Sunda yang disimpan dalam bentuk PDF.

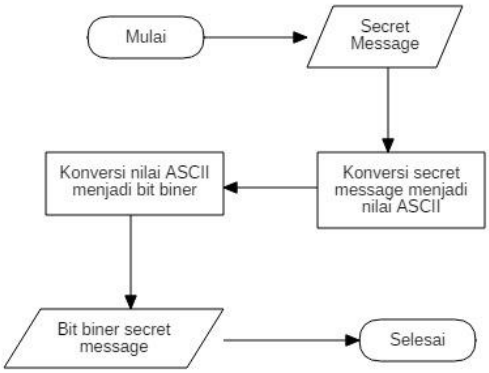
Modul ekstraksi pada sistem ini akan menampilkan tampilan antarmuka untuk pemilihan opsi geser dan berkas PDF dari *stego text* yang diunggah. Modul ekstraksi akan melakukan ekstraksi *secret message* dari berkas PDF yang diunggah. Hasil keluaran dari modul ekstraksi adalah *secret message* yang dapat dilihat pada tampilan antarmuka modul ekstraksi.

3.2.1. Deskripsi Modul *Embedding*

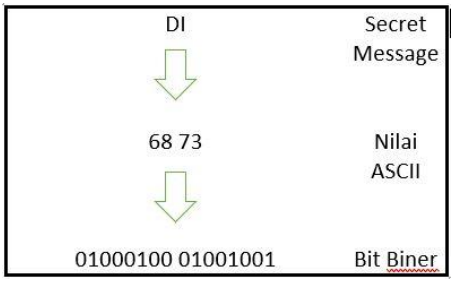
Pada tugas akhir ini, modul *embedding* bertujuan untuk memproses penyisipan *secret message* untuk menyembunyikannya ke dalam *cover text*. Pada modul *embedding*, data masukan yang diterima berupa *secret message*, opsi geser dan *cover text*. Proses yang dilakukan dalam modul *embedding* meliputi tahap pemasukan *secret message*, konversi *secret message*, pembacaan *cover text*, penghitungan jumlah baris untuk penyisipan *secret message* sesuai opsi geser yang dipilih, penyisipan *secret message* melalui pergeseran baris kalimat, dan penyimpanan *stego text*.

Proses pada modul *embedding* dimulai dengan memasukkan *secret message*. Setiap huruf dalam *secret message* yang sudah dimasukkan akan dikonversi dalam bentuk nilai ASCII. Dari nilai ASCII, proses akan dilanjutkan dengan pengubahan nilai ASCII ke dalam bentuk bit biner. Diagram alir proses konversi *secret message* seperti ditunjukkan pada Gambar 3.2.

Pada Gambar 3.3 ditunjukkan contoh konversi *secret message* yang berisi pesan “DI”. Setiap karakter pada *secret message* yang sudah diterima kemudian diubah bentuknya dengan dikonversi ke dalam nilai ASCII menjadi “68 73”. Nilai ASCII “68 73” kemudian diolah lagi masing-masing nilainya menjadi bentuk bit biner sehingga menghasilkan bit biner “01000100 01001001”.

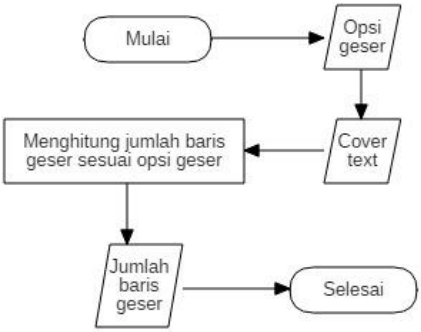


Gambar 3.2 Diagram alir konversi *secret message*



Gambar 3.3 Contoh konversi *secret message* ke dalam nilai ASCII hingga menjadi bit biner

Proses selanjutnya yang dilakukan oleh sistem steganografi adalah pembacaan *cover text* dalam bentuk file docx yang telah diunggah. Proses pembacaan *cover text* dilakukan dengan bantuan *library* Apache POI. Selanjutnya, proses yang dilakukan adalah penghitungan jumlah baris untuk keperluan penyisipan *secret message*. Diagram alir pembacaan *cover text* untuk menghitung jumlah baris bergeser sesuai opsi geser seperti ditunjukkan pada gambar Gambar 3.4.



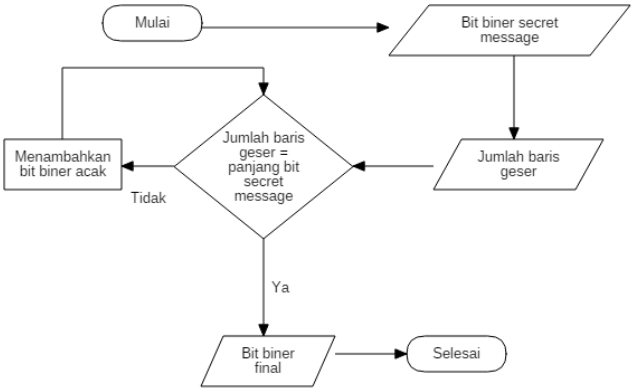
Gambar 3.4 Gambar diagram alir pembacaan *cover text*

26	Jumlah baris
IYA	Secret Message
73 89 65	Nilai ASCII
01001001 01011001 01000001	Bit Biner
(Panjang bit biner = 24)	
01001001 01011001 01000001 + 01	Bit Biner Acak
(Panjang bit biner *secret message* + bit biner acak = 26)	
01001001010110010100000101	Bit Biner Final

Gambar 3.5 Contoh penambahan bit biner acak pada bit biner hasil konversi *secret message*

Jumlah baris yang sudah terhitung akan digunakan sebagai parameter untuk proses penyisipan *secret message* pada *cover text*. Bilamana jumlah panjang bit biner hasil konversi *secret message* melebihi jumlah baris untuk penyisipan, maka proses penyisipan tidak bisa dilakukan. Sebaliknya, bila jumlah panjang bit biner hasil konversi sama dengan jumah baris untuk penyisipan, maka proses penyisipan akan dilakukan seperti pada Gambar 3.5. Proses penambahan bit biner acak akan dilakukan bilamana jumlah panjang bit biner hasil konversi *secret message* lebih sedikit dibanding jumlah baris yang tersedia untuk proses penyisipan *secret message* sampai kondisi dimana jumlah keduanya sama.

Diagram alir penambahan bit biner acak seperti ditunjukkan pada Gambar 3.6.



Gambar 3.6 Diagram alir penambahan bit biner acak

Jumlah dari baris kalimat yang digeser untuk penyisipan *secret message* bergantung pada opsi geser yang dipilih. Pada opsi geser baris genap, baris kalimat yang bisa digeser hanyalah baris pada urutan genap yang memiliki baris kalimat urutan ganjil yang berada di atas maupun di bawahnya. Baris kalimat urutan genap yang menjadi penutup dokumen tidak akan terhitung.

Gambar 3.7 Penghitungan baris yang berubah posisinya berdasar opsi geser baris genap

Contoh pergeseran baris dengan baris genap seperti ditunjukkan pada Gambar 3.7. Terdapat 6 baris, akan tetapi yang bergeser adalah baris ke-2 dan ke-4 karena berada diantara baris ganjil yang posisinya tetap. Sementara itu, baris ke-6 merupakan baris terakhir dalam halaman dan tidak berada diantara dua baris ganjil yang posisinya tetap, maka baris ke-6 pun tidak bergeser dan posisinya tetap.

Pada opsi baris geser 1-5, baris kalimat ke-1 dan ke-5 tidak akan mengalami pergeseran posisi, posisi tetap tanpa pergeseran berlaku juga untuk baris berjarak 5 urutan dari baris pertama dan baris kelima. Tiga baris diantara baris-baris yang posisinya tetap akan mengalami pergeseran dan hanya akan bergeser apabila berada diantara dua baris yang posisinya tetap. Jumlah baris-baris itu akan diakumulasikan sebagai jumlah baris yang tersedia untuk proses penyisipan *secret message* pada opsi geser 1-5. Urutan baris-baris kalimat yang tidak berada diantara dua baris yang tetap posisinya tidak akan mengalami pergeseran.

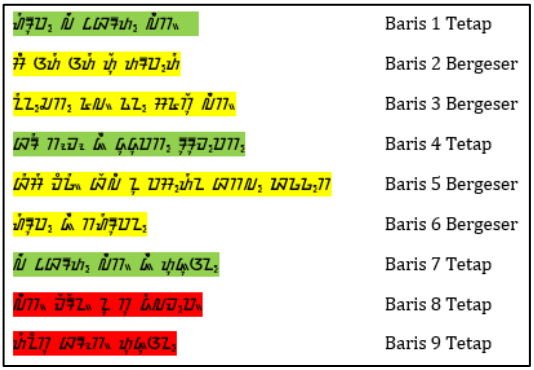
Gambar 3.8 Penghitungan baris yang berubah posisinya berdasar opsi geser baris 1-5

Contoh pergeseran baris sesuai opsi geser baris 1-5 seperti ditunjukkan pada Gambar 3.8. Baris ke-1 dan ke-5 merupakan baris yang posisinya tetap, sedangkan diantara baris ke-1 dan ke-5

yang posisinya tetap terdapat baris ke-2 sampai ke-4, sehingga tiga baris diantara baris ke-1 dan ke-5 akan mengalami pergeseran.

Baris ke-6 merupakan baris kelipatan lima dari baris ke-1 sehingga posisinya pun tetap. Pada contoh ini, hanya terdapat 9 baris dalam halaman, artinya tidak ada baris ke-10 yang menjadi baris kelipatan 5 dari baris ke-5 yang posisinya akan tetap. Akhirnya, baris ke-7 sampai ke-9 tidak bergeser dan tetap posisinya karena tidak berada diantara dua baris yang posisinya tetap.

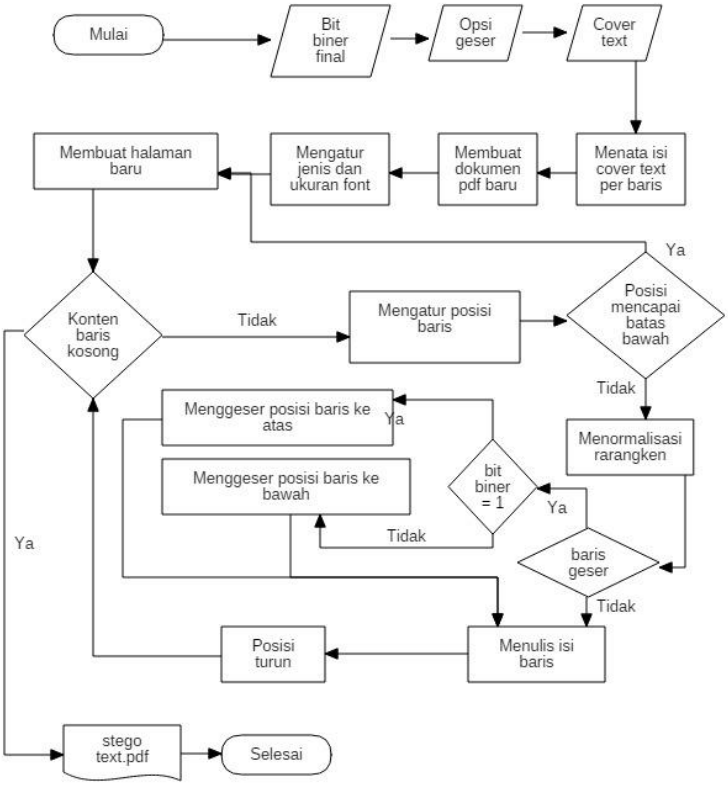
Opsi geser baris 1-4-7 menjadikan baris kalimat ke-1, ke-4, dan ke-7 sebagai baris yang posisinya tetap. Baris kelipatan ketiga dihitung sejak baris kalimat ketujuh juga akan menjadi baris yang posisinya tetap sehingga jarak antara baris-baris kalimat yang tetap adalah 3 baris. Urutan baris-baris kalimat yang berada diantara dua baris yang tetap posisinya akan mengalami pergeseran. Jumlah baris-baris itu akan diakumulasikan sebagai jumlah baris yang tersedia untuk proses penyisipan *secret message* pada opsi geser 1-4-7. Urutan baris-baris kalimat yang tidak berada diantara dua baris yang tetap posisinya tidak akan mengalami pergeseran.



Gambar 3.9 Penghitungan baris yang bergeser posisinya berdasar opsi geser baris 1-4-7

Contoh pergeseran dengan opsi geser baris 1-4-7 seperti digambarkan pada Gambar 3.9. Baris ke-1 dan ke-4 merupakan

baris tetap. Karena berada diantara baris yang posisinya tetap, maka baris ke-2 dan ke-3 akan mengalami pergeseran. Posisi baris ke-7 adalah tetap karena merupakan baris kelipatan urutan 3 dari baris ke-1. Oleh karena itu, baris ke-5 dan ke-6 akan mengalami pergeseran, sebab berada diantara baris ke-4 dan ke-7 yang posisinya tetap. Seperti pada opsi geser baris genap dan baris 1-5, karena tidak berada diantara dua baris yang posisinya tetap, maka baris ke-8 dan ke-9 juga tidak akan mengalami pergeseran dan menjadi tetap posisinya.



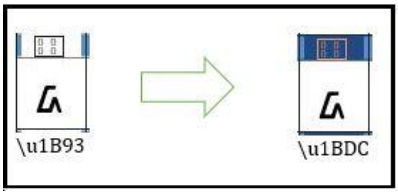
Gambar 3.10 Gambar diagram alir pergeseran baris

Proses penyisipan *secret message* berlangsung dengan mengganti *Unicode* dari setiap karakter yang terdapat pada baris yang digeser sesuai dengan opsi geser. Proses penyisipan *secret message* berlangsung di dalam proses penulisan dokumen *stego text* dalam berkas PDF. Diagram alir pergeseran baris seperti ditunjukkan pada Gambar 3.10.

Setiap karakter pada baris yang bergeser akan melalui proses normalisasi posisi rarangkén terlebih dahulu sebelum melalui proses penggantian *Unicode*. Setiap karakter yang ada pada baris yang tidak bergeser posisinya juga akan melalui proses normalisasi posisi rarangkén. Proses normalisasi perlu dilakukan karena *library* Apache PDFBox tidak mendukung OpenType *Layout* yang terdapat pada *font* Sundanese Unicode 2013. Oleh karena itu, sistem akan menukar *Unicode* dari rarangkén sebelumnya menjadi sebuah *Unicode* baru seperti ditunjukkan pada Gambar 3.11.

Gambar 3.11 Contoh normalisasi posisi rarangkén

Perubahan posisi pada baris kalimat yang digeser akan menyesuaikan bit biner hasil konversi *secret message*. Apabila bit biner menunjukkan angka 1, maka posisi baris kalimat akan bergeser secara vertikal ke atas. Hal ini dilakukan dengan mengganti *Unicode* dari setiap karakter pada baris menjadi *Unicode* baru yang telah dimodifikasi. Sementara untuk kondisi bit biner menunjukkan angka 0, posisi baris kalimat akan bergeser secara vertikal ke bawah. Hal ini dilakukan dengan mengganti *Unicode* dari setiap karakter pada baris menjadi *Unicode* baru yang telah dimodifikasi seperti ditunjukkan pada Gambar 3.12.



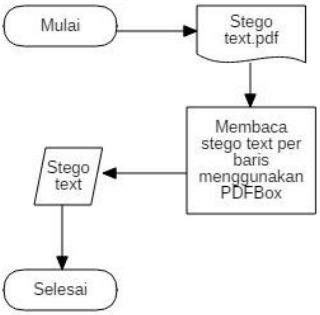
Gambar 3.12 Contoh penukaran blok *Unicode* karakter pada baris yang bergeser

Setiap baris tetap yang sudah melalui proses normalisasi rarangkén maupun baris bergeser yang sudah melalui proses penyisipan *secret message* akan ditulis sesuai urutannya oleh sistem ke dalam berkas dokumen berbentuk PDF. Berkas dokumen PDF ini merupakan berkas yang berisi *stego text* yang sudah disisipi *secret message* dan merupakan hasil keluaran dari modul *embedding*.

3.2.2. Deskripsi Modul Ekstraksi

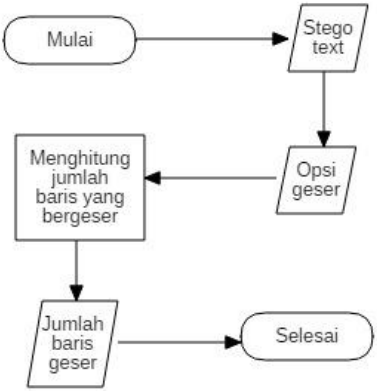
Modul ekstraksi memiliki tujuan untuk mengembalikan *secret message* dari *stego text*. Data masukan pada modul ekstraksi berupa opsi geser dan *stego text* dalam bentuk berkas PDF yang diunggah. Gambaran umum proses kerja dalam modul ekstraksi berupa penerimaan dan pembacaan *stego text*, penghitungan panjang bit *secret message* dan ekstraksi *stego text* untuk mendapatkan *secret message*.

Proses pada modul ekstraksi diawali dengan pembacaan *stego text* yang telah diterima. Proses pembacaan akan menghasilkan string Aksara Sunda dengan bantuan *library* Apache PDFBox. Diagram alir pembacaan *stego text* seperti ditunjukkan pada Gambar 3.13.



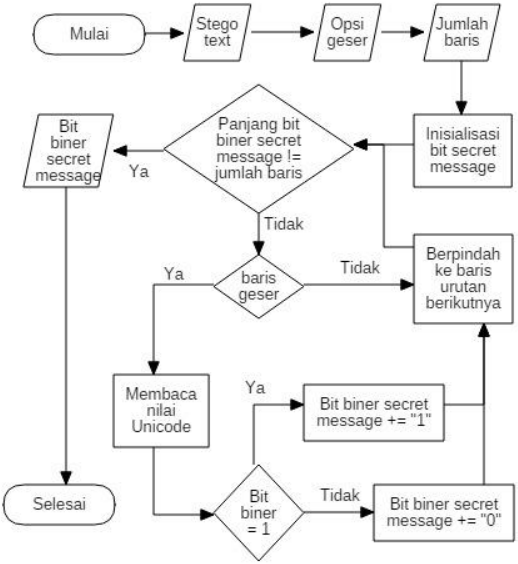
Gambar 3.13 Diagram alir pembacaan *stego text*

Stego text yang sudah dibaca akan digunakan untuk menghitung jumlah baris yang bergeser sesuai dengan opsi geser. Diagram alir penghitungan baris sesuai opsi geser yang dipilih seperti ditunjukkan pada Gambar 3.14.



Gambar 3.14 Diagram alir penghitungan baris yang bergeser pada *stego text*

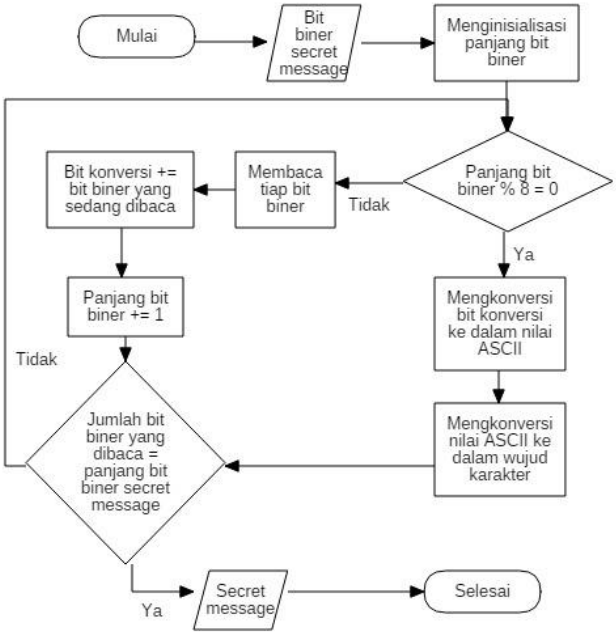
Setiap baris yang bergeser akan dicek nilai *Unicode*-nya. Dari pengecekan baris yang bergeser, angka didapatkan nilai “1” ataupun “0” sesuai pengecekan *Unicode* yang dilakukan. Nilai “1” atau “0” inilah yang kemudian akan digabungkan menjadi bit biner dari *secret message*. Opsi geser yang dipilih akan memengaruhi jumlah baris yang dianggap bergeser, sehingga apabila panjang bit biner sudah sama dengan jumlah baris yang dihitung bergeser maka proses pengecekan *Unicode* selesai. Diagram alir penghitungan panjang bit biner *secret message* seperti ditunjukkan pada Gambar 3.15



Gambar 3.15 Diagram alir penghitungan panjang *secret message*

Bit biner yang sudah didapatkan kemudian dikonversikan ke dalam nilai ASCII untuk setiap 8 bit. Nilai ASCII akan dikembalikan dalam bentuk teks sebagai *secret message* yang dikembalikan dari *stego text* untuk ditampilkan pada antarmuka

modul ekstraksi. Diagram alir ekstraksi *secret message* pada *stego text* seperti ditunjukkan pada Gambar 3.16.

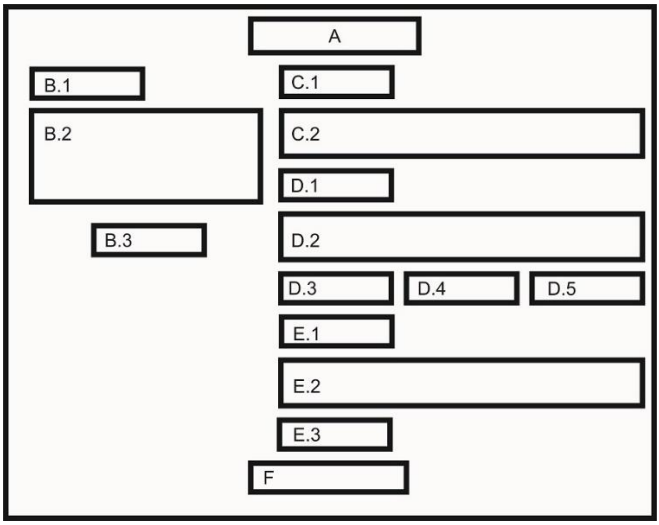


Gambar 3.16 Diagram alir ekstraksi *secret message* pada *stego text*

3.3. Perancangan Antarmuka

Pada tugas akhir ini, antarmuka dibuat untuk memudahkan penggunaan sistem oleh pengguna. Antarmuka sistem steganografi yang dibangung akan memiliki dua modul, modul *embedding* dan modul ekstraksi. Antarmuka sistem dibangun dengan menggunakan Jframe *class* pada Netbeans *IDE* 8.0.2. Rincian tampilan dari antarmuka yang dibangun dapat dilihat dalam sub-bab di bawah.

3.3.1. Antarmuka Modul *Embedding*



Gambar 3.17 Desain antarmuka modul *embedding*

Desain antarmuka dari modul *embedding* digambarkan pada Gambar 3.17. Rincian komponen yang terdapat dalam desain modul *embedding* terdiri dari :

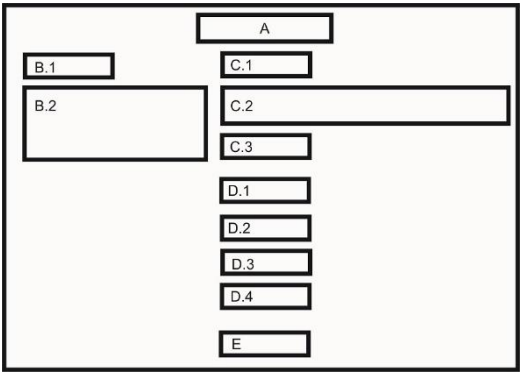
A. Bagian A berupa label yang menunjukkan nama dari sistem.
B. Bagian B adalah bagian untuk memasukkan *secret message* ke dalam sistem. B.1 adalah label yang menandakan *secret message*. B.2 adalah kolom masukan untuk *secret message*. B.3 adalah tombol untuk memasukkan *secret message* ke dalam sistem.
C. Bagian C adalah bagian yang menampilkan hasil konversi *secret message* ke dalam bit biner. C.1 adalah label penanda hasil konversi. C.2 adalah kolom yang menampilkan bit biner hasil konversi dari *secret message.*
D. Bagian D adalah bagian untuk pemilihan opsi geser. D.1 adalah label penanda jumlah baris yang tersedia sesuai

pemilihan opsi geser. D.2 adalah kolom yang menampilkan jumlah baris yang tersedia untuk penyisipan *secret message*. D.3 adalah tombol untuk memilih opsi geser baris genap. D.4 adalah tombol untuk memilih opsi geser baris 1-5. D.5 adalah tombol untuk memilih opsi baris geser 1-4-7.

E. Bagian E adalah bagian untuk mengunggah *cover text*. E.1 adalah label penanda *cover text*. E.2 adalah kolom yang berisi lokasi berkas *cover text* yang diunggah. E.3 adalah tombol untuk mengunggah *cover text* ke dalam sistem.

F. Bagian F adalah tombol untuk memulai proses penyisipan *secret message* ke dalam *cover text* pada modul *embedding*.

3.3.2. Antarmuka Modul Ekstraksi



Gambar 3.18 Desain antarmuka modul ekstraksi

Desain antarmuka dari modul ekstraksi digambarkan pada Gambar 3.18. Rincian komponen yang terdapat dalam desain modul ekstraksi terdiri dari :

A. Bagian A adalah label yang menunjukkan nama dari sistem.

B. Bagian B adalah bagian yang menunjukkan *secret message* yang didapatkan dari modul ekstraksi. B.1 adalah label penanda *secret message*. B.2 adalah kolom yang menampilkan *secret message* yang didapatkan.

C. Bagian C adalah bagian yang menunjukkan *stego text*. C.1 adalah label penanda *stego text*. C.2 adalah kolom yang menampilkan lokasi berkas *stego text* yang diunggah. C.3 adalah tombol untuk mengunggah *stego text* ke dalam sistem.

D. Bagian D adalah bagian untuk memilih opsi geser. D.1 adalah label penanda pilihan opsi geser. D.2 adalah *radio button* untuk memilih opsi geser baris genap. D.3 adalah *radio button* untuk memilih opsi geser baris 1-5. D.4 adalah *radio button* untuk memilih opsi geser baris 1-4-7.

E. Bagian E adalah tombol untuk memulai proses ekstraksi *secret message* dari *stego text* yang diunggah berdasar opsi geser yang dipilih.

BAB IV
IMPLEMENTASI

Bab ini akan menjelaskan tentang implementasi sistem steganografi teks pada Aksara Sunda dengan pendekatan *line shift coding* yang di dalamnya akan dibahas lingkungan pengembangan sistem dan implementasi dari modifikasi *font*, modul *embedding*, dan modul ekstraksi. Proses implementasi pada bab ini mengacu pada rancangan perangkat yang telah dilakukan pada bab sebelumnya.

4.1. Lingkungan Implementasi

Subbab ini menjelaskan tentang lingkungan implementasi sistem steganografi teks pada Aksara Sunda dengan pendekatan *line shift coding* yang dibangun. Lingkungan selama proses implementasi sistem steganografi teks pada Aksara Sunda dengan pendekatan *line shift coding* dapat dilihat pada Tabel 4.1.

Tabel 4.1 Lingkungan Implementasi Sistem

Perangkat	Jenis	Spesifikasi
Perangkat Keras	Prosesor	Intel® Core™ i3-6006U CPU @ 2.00 GHz 2.00 GHz
	RAM	4 GB
Perangkat Lunak	Sistem Operasi	Microsoft Windows 10 Home Single Language
	Bahasa Pemrograman	Java
	IDE	Netbeans 8.0.2
	Font Editor	FontForge Version 31-07-2017

4.2. Implementasi Modifikasi *Font*

Subbab ini menjelaskan tentang modifikasi pada *font* Sundanese Unicode 2013. Proses modifikasi dilakukan dengan bantuan FontForge version 31-07-2017 sebagai *font editor*. Beberapa modifikasi yang dilakukan adalah penambahan blok *Unicode* baru untuk rarangkén kombinasi yang mempunyai pasangan yang berada dalam satu posisi. Modifikasi rarangkén kombinasi seperti ditunjukkan pada Tabel 4.2.

Tabel 4.2 Implementasi Modifikasi Rarangkén Kombinasi

BLOK UNICODE NORMAL				
\u1bc0	\u1bc1	\u1bc2	\u1bc3	\u1bc4
				
\u1bc5	\u1bc6	\u1bc7	\u1bc8	
				

Rarangkén kombinasi hasil modifikasi akan menjadi satu blok dengan karakter lain dalam blok normal yang tidak mengalami pergeseran posisi. Blok *Unicode* normal seperti ditunjukkan pada Tabel 4.3.

Tabel 4.3 Implementasi Blok *Unicode* Normal

BLOK UNICODE NORMAL						
\u1b80	\u1b81	\u1b82	\u1b83	\u1b84	\u1b85	\u1b86
						
\u1b87	\u1b88	\u1b89	\u1b8a	\u1b8b	\u1b8c	\u1b8d
						
\u1b8e	\u1b8f	\u1b90	\uib91	\u1b92	\u1b93	\u1b94

						
\u1b95	\u1b96	\u1b97	\u1b98	\u1b99	\u1b9a	\u1b9b
						
\u1b9c	\u1b9d	\u1b9e	\u1b9f	\u1ba0	\u1ba1	\u1ba2
						
\u1ba3	\u1ba4	\u1ba5	\u1ba6	\u1ba7	\u1ba8	\u1ba9
						
\u1baa	\u1bae	\u1baf	\u1bb0	\u1bb1	\u1bb2	\u1bb3
						
\u1bb4	\u1bb5	\u1bb6	\u1bb7	\u1bb8	\u1bb9	\u1bc0
						
\u1bc1	\u1bc2	\u1bc3	\u1bc4	\u1bc5	\u1bc6	\u1bc7
\u1bc8						

Modifikasi lain yang dilakukan pada *font* Sundanese Unicode 2013 adalah penambahan blok Unicode baru untuk menampung replika dari *font* yang telah digeser posisinya untuk keperluan penggeseran sesuai pendekatan *line shift coding*. Karakter yang mengalami pergeseran ke bawah disimpan dalam blok-blok baru seperti ditunjukkan pada Tabel 4.4.

Tabel 4.4 Implementasi Blok *Unicode* yang Mengalami Pergeseran Posisi Ke Bawah

BLOK UNICODE GESER KE BAWAH						
\u1bc9	\u1bca	\u1bcb	\u1bcc	\u1bcd	\u1bce	\u1bcf
						
\u1bd0	\u1bd1	\u1bd2	\u1bd3	\u1bd4	\u1bd5	\u1bd6
						
\u1bd7	\u1bd8	\u1bd9	\u1bda	\u1bdb	\u1bdc	\u1bdd
						
\u1bde	\u1bdf	\u1be0	\u1be1	\u1be2	\u1be3	\u1be4
						
\u1be5	\u1be6	\u1be7	\u1be8	\u1be9	\u1bea	\u1beb
						
\u1bec	\u1bed	\u1bee	\u1bef	\u1bf0	\u1bf1	\u1bf2
						
\u1bf3	\u1bf7	\u1bf8	\u1bf9	\u1bfa	\u1bfb	\u1bfc
\u1bfd	\u1bfe	\u1bff	\u1c00	\u1c01	\u1c02	\u1b09
\u1c0a	\u1c0b	\u1c0c	\u1c0d	\u1c0e	\u1c0f	\u1c10

\u1c11						
						

Sedangkan untuk karakter yang mengalami pergeseran posisi ke atas juga disimpan dalam blok lain seperti ditunjukkan pada Tabel 4.5.

Tabel 4.5 Implementasi Blok *Unicode* yang Mengalami Pergeseran Posisi Ke Atas

BLOK UNICODE GESER KE ATAS						
\u1c12	\u1c13	\u1c14	\u1c15	\u1c16	\u1c17	\u1c18
						
\u1c19	\u1c1a	\u1c1b	\u1c1c	\u1c1d	\u1c1e	\u1c1f
						
\u1c20	\u1c21	\u1c22	\u1c23	\u1c24	\u1c25	\u1c26
						
\u1c27	\u1c28	\u1c29	\u1c2a	\u1c2b	\u1c2c	\u1c2d
						
\u1c2e	\u1c2f	\u1c30	\u1c31	\u1c32	\u1c33	\u1c34
						
\u1c35	\u1c36	\u1c37	\u1c38	\u1c39	\u1c3a	\u1c3b
						

\u1c3c	\u1c40	\u1c41	\u1c42	\u1c43	\u1c44	\u1c45
						
\u1c46	\u1c47	\u1c48	\u1c49	\u1c4a	\u1c4b	\u1c52
						
\u1c53	\u1c54	\u1c55	\u1c56	\u1c57	\u1c58	\u1c59
						
\u1c5a						
						

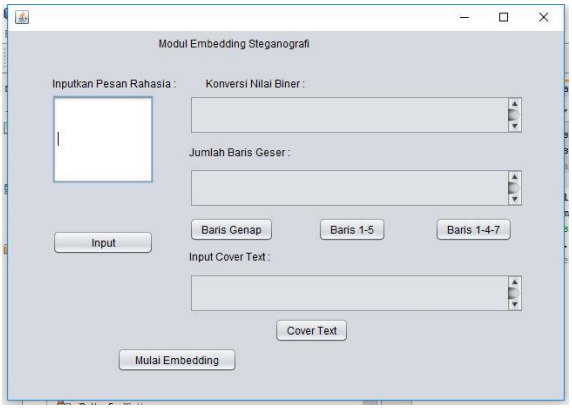
Pungtuasi (tanda baca) pada *font* Sundanese Unicode tidak mengalami duplikasi untuk pembuatan replikanya. Jadi, pada proses penyisipan *secret message* karakter yang mewakili pungtuasi tidak akan mengalami pergeseran.

4.3. Implementasi Antarmuka Modul *Embedding*

Tampilan antarmuka pada modul *embedding* dibangun sesuai dengan desain yang telah dibuat pada bab analisis dan perancangan sistem. Hasil implementasi antarmuka pada modul *embedding* seperti ditunjukkan pada Gambar 4.1.

Implementasi antarmuka pada modul *embedding* dibangun pada lingkungan bahasa pemrograman Java dengan Netbeans 8.0.2 sebagai *IDE*. Tampilan antarmuka dibuat menggunakan JForm yang ada pada *IDE* Netbeans.

Dalam antarmuka ini pengguna dapat memasukkan *secret message* lalu mengubahnya menjadi bentuk biner. Kemudian mengunggah berkas *cover text* berisi Aksara Sunda dan memilih opsi geser. Setelah itu pengguna dapat melakukan proses penyisipan *secret message*.

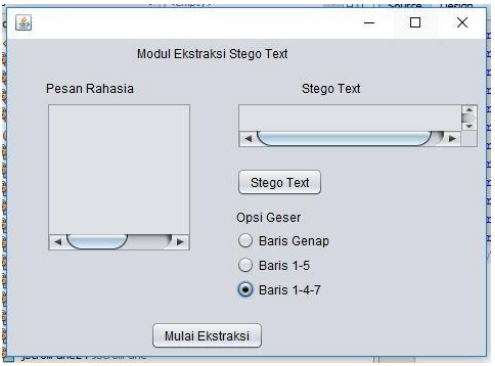


Gambar 4.1 Tampilan Antarmuka Modul Embedding

4.4. Implementasi Antarmuka Modul Ekstraksi

Tampilan antarmuka pada modul ekstraksi dibangun sesuai dengan desain yang telah dibuat pada bab analisis dan perancangan sistem. Hasil implementasi antarmuka pada modul ekstraksi seperti ditunjukkan pada Gambar 4.2.

Implementasi antarmuka pada modul ekstraksi dibangun pada lingkungan bahasa pemrograman Java dengan Netbeans 8.0.2 sebagai *IDE*. Tampilan antarmuka dibuat menggunakan JForm yang ada pada *IDE* Netbeans.



Gambar 4.2 Tampilan Antarmuka Modul Ekstraksi

Dalam antarmuka ini pengguna dapat mengunggah berkas *stego text* yang akan diekstraksi. Kemudian pengguna dapat memilih opsi geser dan sehingga sistem dapat melakukan proses ekstraksi untuk mendapatkan kembali *secret message* dari *stego text*.

4.5. Implementasi Modul *Embedding*

Secara umum, proses yang terjadi pada modul *embedding* terdiri dari konversi *secret message*, pembacaan *cover text* dan penghitungan jumlah baris geser, normalisasi rarangkén, pergeseran baris, dan penyimpanan *stego text*. Penjelasan tahapan proses dirincikan pada masing-masing subbab.

4.5.1. Implementasi Konversi *Secret Message*

Proses konversi *secret message* diawali dengan memasukkan *secret message*. Data masukan berupa *secret message* ini diubah menjadi bentuk bit biner dengan memanggil fungsi konversipesan. *Pseudocode* proses konversi *secret message* seperti ditunjukkan Gambar 4.3. *Pseudocode* fungsi konversipesan seperti ditunjukkan pada gambar Gambar 4.4.

```
1  READ secret message
2  var <-- ""
3  pesanrahasia <--"
4  jmlbiner <-- 0
5  var <-- secret message
6  pesanrahasia <-- CALL konversipesan with
   var RETURNING pesankonversi
7  jmlbiner <-- CALL jumlahbinerpsnrhsia
   with pesanrahasia RETURNING jumlahbiner
8  OUT pesanrahasia
```

Gambar 4.3 Implementasi proses konversi *secret message*

```
FUNCTION konversipesan(pesanrahasia):
bytes <-- []
pesankonversi <-- ""
binary <-- ""
byte <-- pesanrahasia.getBytes()
for byte b in bytes
    val <-- b
    for 0 < i < 8
        IF val & 128 == 0 THEN
            val <-- 0
        ELSE val <-- 1
        binary <-- binary + val.toString
        val <<= 1
pesankonversi <-- binary
RETURN pesankonversi
ENDFUNCTION
```

Gambar 4.4 Implementasi fungsi konversipesan

Hasil fungsi konversipesan disimpan dalam variabel pesankonversi (*StringBuilder*), sistem kemudian akan menggunakan variabel pesankonversi sebagai parameter fungsi jumlahbinerpsnrhsia untuk menghitung jumlah panjang bit biner. *Pseudocode* fungsi jumlahbinerpsnrhsia seperti pada Gambar 4.5.

```
FUNCTION jumlahbinerpsnrhsia
(pesanrahasia):
jumlahbiner <-- 0
jumlahbiner <-- pesanrahasia.length
RETURN jumlahbiner
ENDFUNCTION
```

Gambar 4.5 Implementasi fungsi jumlahbinerpsnrhsia

Hasil dari fungsi jumlahbinerpsnrhsia disimpan dalam variabel jumlahbiner (*int*) yang berisi jumlah panjang bit biner dari pesan rahasia yang telah dikonversi.

4.5.2. Implementasi Pembacaan *Cover Text*

Proses pembacaan *cover text* dilakukan dengan memanggil fungsi bacadocx dengan lokasi (*path*) dari file *cover text* yang dipilih sebagai parameter. Fungsi bacadocx akan menghasilkan variabel *string* isicovertext yang berisi teks dari *cover text* yang berhasil dibaca.

Setelah membaca *cover text* sistem akan memanggil fungsi bacapdf_hitungbaris untuk menghitung jumlah baris dalam *cover*

text yang bisa digeser. Pada fungsi bacapdf_hitungbaris *string* hasil fungsi bacadocx dan opsi geser digunakan sebagai parameter fungsi. *Pseudocode* proses pembacaan cover text dan penghitungan baris geser seperti pada Gambar 4.6.

```
1  READ path
2  jumlahbarisgeser <-- 0
3  covertext <-- ""
4  covertext <-- CALL bacadocx with path RETURNING
   isicovertext
5  jumlahbarisgeser <-- CALL bacapdf_hitungbaris with
   covertext, opsigeser
6  OUT jumlahbarisgeser
```

Gambar 4.6 Implementasi proses pembacaan *cover text*

Fungsi bacadocx akan membaca file *cover text* yang dipilih. Fungsi bacadocx memiliki parameter path (*string*) yang berisi lokasi file *cover text*. File *cover text* memiliki ekstensi docx. Sistem akan membaca isi dari file per paragraf. Proses pembacaan terus dilakukan berulang sampai paragraf terakhir. Hasil pembacaan disimpan pada covertext (*string*). Rincian fungsi bacadocx seperti pada Gambar 4.7.

```
1  FUNCTION bacadocx(path):
2  tempstring <-- ""
3  isicovertext <-- null
4  eol <-- "\r\n"
5  docx <-- new XWPFDocument from path
6  paragraphList [] <-- docx.getParagraphs()
7  for XWPFParagraph paragraph in paragraphList
8      tempstring <-- tempstring + paragraph.getText() + eol
9      isicovertext <-- tempstring.toString
10 RETURN isicovertext.trim
11 ENDFUNCTION
```

Gambar 4.7 Implementasi fungsi bacadocx

Fungsi bacapdf_hitungbaris digunakan untuk menghitung jumlah baris pada *cover text* yang bisa digeser. Fungsi bacapdf_hitungbaris memiliki parameter covertext (*string*) dan opsigeser (*integer*).Penghitungan baris dilakukan dengan membuat dokumen pdf baru. Sistem menginisiasi pengaturan dokumen pdf seperti ukuran *font*¸ jenis *font*, dan nilai margin dokumen pdf. Setelah menginisiasi pengaturan dokumen pdf, sistem memanggil fungsi stringperbaris untuk membagi *string* covertext menjadi teks

per baris. Baris-baris hasil fungsi stringperbaris inilah yang dihitung sebagai baris-baris dalam *cover text*.

```
FUNCTION bacapdf_hitungbaris(covertext,opsigeser):
doc <-- null
jmlbarisopsi, jmlbaris, barisgeser <-- 0
doc <-- new PDDocument
page <-- new PDPage size A4
add this page to doc
content <-- new PDPageContentStream in doc, page
pdfFont <-- load font SundaneseUnicode2013_modifikasi.ttf
leading <-- 2.1f * 11
mediabox <-- page.getMediaBox
margin,marginplus,marginbawah <-- 72
width <-- mediabox.getWidth() - 2*margin
startX <-- mediabox.getLowerLeftX() + margin
startY <-- mediabox.getUpperRightY() - marginplus
textNL <-- covertext.replace("\t","")
lines <-- CALL stringperbaris with textNL, fontSize, width, pdfFont RETURNING lines
listsize <-- lines.size
beginText in content
setFont with pdfFont, size=11 in content
newLineAtOffset with startX, startY in content
currentY <-- startY
FOR line in lines
        currentY <-- currentY - leading
        IF currentY <= marginbawah THEN
                endText in content
                content.close()
                new_Page <-- new PDPage size A4
                add this new_Page to doc
                content <-- new PDPageContentStream in doc, new_Page
                beginText in  content
                setFont with pdfFont, size=11 in content
                newLineAtOffset with startX, startY in content
                currentY <-- startY
                IF opsigeser == 1 and jmlbaris%2 == 0 THEN
                        barisgeser <-- barisgeser - 1
                jmlbarisopsi <-- jmlbaris
                jmlbaris <-- 0
        jmlbaris <-- jmlbaris + 1
        barisgeser <-- barisgeser + CALL barisgeser_opsigeser with opsigeser, jmlbaris RETURNING barisgeser
        showText with line in content
        newLineAtOffset with 0, currentY-leading
endText in content
close content
IF opsigeser == 1 and jmlbaris%2 == 0 THEN
        barisgeser--;
CALL call_baris with barisgeser
save doc to path
IF doc != null THEN
        close doc
RETURN true
ENDFUNCTION
```

Gambar 4.8 Implementasi fungsi bacapdf_hitungbaris

Sistem kemudian memanggil fungsi barisgeser_opsigeser untuk menghitung jumlah baris geser pada *cover text*. Setelah mendapatkan jumlah baris geser, sistem memanggil fungsi call_baris untuk menyimpan jumlah baris geser dan menampilkannya di antarmuka modul *embedding*. Rincian fungsi bacapdf_hitungbaris seperti ditunjukkan pada Gambar 4.8.

Di dalam fungsi bacapdf_hitungbaris, sistem memanggil fungsi stringperbaris. Pada fungsi stringperbaris, sistem melakukan *split* pada covertext yang ada pada parameter textNL (*string*) dimana panjang dari *split* ditentukan berdasar ukuran *font*, jenis *font*, dan lebar dokumen. Pseudocode fungsi stringperbaris seperti ditunjukkan pada Gambar 4.9.

```
FUNCTION stringperbaris(textNL, fontSize, width, pdfFont):
lines <-- []
FOR text in textNL.split("\\r?\\n")
    lastSpace <-- -1
    WHILE text.length() > 0)
        spaceIndex <-- text.indexOf(' ', lastSpace + 1)
        IF spaceIndex < 0 THEN
            spaceIndex <-- text.length()
        subString <-- text.substring(0, spaceIndex)
        size <-- fontSize *
        pdfFont.getStringWidth(subString) / 1000
        IF size > width THEN
            IF lastSpace < 0 THEN
                lastSpace <-- spaceIndex
            subString <-- text.substring(0, lastSpace)
            add lines with subString
            text <-- text.substring(lastSpace).trim
            lastSpace <-- -1
        ELSE IF spaceIndex == text.length() THEN
            add lines with text
            text <-- ""
        ELSE
            lastSpace <-- spaceIndex
RETURN lines
ENDFUNCTION
```

Gambar 4.9 Implementasi fungsi stringperbaris

Sistem juga memanggil fungsi barisgeser_opsigeser di dalam fungsi bacapdf_hitungbaris. Fungsi ini memiliki parameter opsigeser (*integer*) dan jmlbaris (*integer*). Fungsi barisgeser_opsigeser melakukan penghitungan jumlah baris geser dengan melakukan pengecekan pada parameter opsigeser dan jmlbaris yang dimasukkan. Fungsi kemudian mengembalikan nilai

jumlah baris geser pada variabel barisgeser (*integer*). Pseudocode fungsi barisgeser_opsigeser seperti pada Gambar 4.10.

```
1  FUNCTION barisgeser_opsigeser(opsigeser, jmlbaris):
2  barisgeser, b_genap, b_satulima, b_satuempattujuh <-- 0
3  IF opsigeser == 1 THEN
4      b_genap <-- CALL barisgenap with jmlbaris, b_genap
       RETURNING b_genap
5      barisgeser <-- b_genap
6  ELSE IF opsigeser == 2
7      b_satulima <-- CALL barissatulima with jmlbaris,
       b_satulima RETURNING b_satulima
8      barisgeser <-- b_satulima
9  ELSE IF opsigeser == 3
10     b_satuempattujuh <-- CALL barissapatju with
       jmlbaris, b_satuempattujuh RETURNING b_satuempattujuh
11     barisgeser <-- b_satuempattujuh
12 RETURN barisgeser
13 ENDFUNCTION
```

Gambar 4.10 Implementasi fungsi barisgeser_opsigeser

Sistem memanggil fungsi call_baris guna menampilkan jumlah baris yang sudah terhitung. Fungsi ini memiliki parameter jmlbaris (*integer*) yang berisi nilai final jumlah baris geser. Nilai dari parameter jmlbaris disimpan oleh s_jmlbarisgeser (*string*). Nilai final jumlah baris geser kemudian ditampilkan pada antarmuka modul *embedding*. Rincian fungsi call_baris seperti pada Gambar 4.11.

```
1  FUNCTION call_baris(jmlbaris):
2  s_jmlbarisgeser <-- String.valueOf(jmlbaris)
3  OUT s_jmlbarisgeser
4  ENDFUNCTION
```

Gambar 4.11 Implementasi fungsi call_baris

Sistem memanggil fungsi barisgenap di dalam fungsi barisgeser_opsigeser jika parameter opsigeser (*integer*) pada fungsi barisgeser_opsigeser bernilai 1, yang mewakili opsi geser baris genap. Fungsi barisgenap melakukan penghitungan jumlah baris geser pada opsi geser baris genap. Rincian fungsi barisgenap ditunjukkan pada Gambar 4.12.

```
FUNCTION barisgenap (jmlbaris, b_genap):
IF jmlbaris % 2 == 0 THEN
    b_genap <-- b_genap + 1
RETURN b_genap
ENDFUNCTION
```

Gambar 4.12 Implementasi fungsi barisgenap

Sistem memanggil fungsi barissatulima di dalam fungsi barisgeser_opsigeser jika parameter opsigeser (*integer*) pada fungsi barisgeser_opsigeser bernilai 2, yang mewakili opsi geser baris 1-5. Fungsi barissatulima melakukan penghitungan jumlah baris geser pada opsi geser baris 1-5. Rincian fungsi barissatulima seperti ditunjukkan pada Gambar 4.13.

```
FUNCTION barissatulima(jmlbaris, b_satulima):
IF jmlbaris % 5 == 0 THEN
    b_satulima <-- b_satulima + 3
RETURN b_satulima
ENDFUNCTION
```

Gambar 4.13 Implementasi fungsi barissatulima

Sistem memanggil fungsi barissapatjudi dalam fungsi barisgeser_opsigeser jika parameter opsigeser (*integer*) pada fungsi barisgeser_opsigeser bernilai 3, yang mewakili opsi geser baris 1-4-7. Fungsi barissapatju melakukan penghitungan jumlah baris geser pada opsi geser baris 1-4-7. Rincian fungsi barissapatju seperti ditunjukkan pada Gambar 4.14.

```
FUNCTION barissapatju(jmlbaris, b_sapatju):
IF jmlbaris != 1 and jmlbaris % 3 == 1 THEN
    b_sapatju <-- b_sapatju + 2
RETURN b_sapatju
ENDFUNCTION
```

Gambar 4.14 Implementasi fungsi barissapatju

4.5.3. Implementasi Penyimpanan Stego text

Sistem akan melakukan proses penyisipan *secret message* pada *cover text* dengan menggunakan bit biner hasil konversi *secret message* dan isi *cover text* yang sudah didapatkan. Sistem melakukan normalisasi rarangkén dan pergeseran baris saat menuliskan baris-baris dokumen *stego text* menjadi sebuah dokumen pdf. Selanjutnya dokumen pdf *stego text* akan disimpan.

Rincian proses penyimpanan stego text seperti ditunjukkan pada Gambar 4.15. Sistem terlebih dahulu melakukan pembandingan panjang bit biner *secret message* dan jumlah baris geser dengan memanggil fungsi generaterandombiner. Sistem kemudian melakukan proses penyimpanan *stego text* dengan memanggil fungsi embed_simpanstegopdf.

```
1  READ jmlbarisgeser, jmlbiner, pesanrahasia,
   tampungisistring, opsigeser
2  IF jmlbiner < jmlbarisgeser
3      pesanrahasia <-- CALL generaterandombiner with
       jmlbiner, jmlbarisgeser, pesanrahasia RETURNING
       concated_pesanrahasia
4      jmlbiner <-- jmlbarisgeser
5  CALL embed_simpanstegopdf with tampungisistring,
   opsigeser, pesanrahasia
6  OUT stegotext.pdf
```

Gambar 4.15 Implementasi proses penyimpanan *stego text*

```
1  FUNCTION generaterandombiner(jmlbiner, jmlbarisgeser,
   pesanrahasia):
2  concat <-- null
3  rand <-- new Random
4  WHILE jmlbiner < jmlbarisgeser
5      randomNum <-- rand.nextInt(1 - 0 + 1) + 0
6      pesanrahasia <-- pesanrahasia +
       String.valueOf(randomNum)
7      concat <- pesanrahasia
8      jmlbiner <-- jmlbiner + 1
9  RETURN concat
10 ENDFUNCTION
```

Gambar 4.16 Implementasi fungsi generaterandombiner

Rincian fungsi generaterandombiner seperti ditunjukkan pada Gambar 4.16. Penambahan bit biner acak akan dilakukan apabila jumlah baris geser lebih banyak dari bit biner *secret message* yang disisipkan. Penambahan bit biner acak dilakukan sampai panjang bit biner *secret message* sama dengan jumlah baris geser. Fungsi generaterandombiner mengembalikan concat (*string*) yang berisi bit biner final yang sudah panjangnya sama dengan nilai jumlah baris geser.

Sistem melakukan penyimpanan *stego text* dengan memanggil fungsi embed_stegotextpdf. Proses normalisasi rarangkén dan penggeseran baris dilakukan di dalam fungsi ini.

```
FUNCTION embed_simpanstegotextpdf(covertext, opsigeser, binerpesan):
doc <-- null
countshift, jmlbarisopsi, jmlbaris, barisgeser, counteropsidua, counteropsitiga <-- 0
arrayline_temp <-- [5]
a_binerpesan <-- []
a_binerpesan <-- binerpesan.toCharArray()
doc <-- new PDDocument
page <-- new PDPage size A4
add this page to doc
content <-- new PDPageContentStream in doc, page
pdfFont <-- load font SundaneseUnicode2013_modifikasi.ttf
leading <-- 2.1f * 11
mediabox <-- page.getMediaBox
margin,marginplus,marginbawah <-- 72
width <-- mediabox.getWidth() - 2*margin
startX <-- mediabox.getLowerLeftX() + margin
startY <-- mediabox.getUpperRightY() - marginplus
textNL <-- covertext.replace("\t","")
lines <-- CALL stringperbaris with textNL, fontSize, width, pdfFont RETURNING lines
listsize <-- lines.size
beginText in content
setFont with pdfFont, size=11 in content
newLineAtOffset with startX, startY in content
currentY <-- startY
FOR line in lines
    currentY <- currentY - leading
    IF currentY <= marginbawah THEN
        IF opsigeser == 3 THEN
            FOR k=0 to k<counteropsitiga
                showText with arrayline_temp[k] in content
                newLineAtOffset with 0,-leading in content
            counteropsitiga <-- 0
        ELSE IF opsigeser == 2 THEN
            FOR k=0 to k<counteropsidua
                showText with arrayline_temp[k] in content
                newLineAtOffset with 0,-leading in content
            counteropsidua <-- 0
        endText in content
```

Gambar 4.17 Implementasi fungsi embed_stegotextpdf bagian pertama

Seperti ditunjukkan pada gambar Gambar 4.17 fungsi embed_stegotextpdf memiliki parameter covertext (*string*), opsigeser (*integer*) dan binerpesan(*string*). Fungsi embed_stegotextpdf mengubah binerpesan (*string*) yang berisi bit biner *secret message* ke dalam a_binerpesan(*array of char*). Fungsi kemudian menginisiasi pengaturan dokumen pdf dari *stego text* yang akan dibuat. Selanjutnya, fungsi menghilangkan tab yang mungkin terdapat dalam covertext (*string*) dan menyimpan dalam variabel textNL (*string*). Fungsi embed_stegotextpdf kemudian memanggil fungsi stringperbaris untuk mengubah textNL (*string*) menjadi lines (*array of string*) yang berisi baris-baris dari *cover text*. Rincian fungsi stringperbaris ditunjukkan pada Gambar 4.9.

Fungsi embed_stegotextpdf kemudian melakukan penulisan *stego text*. Sebelum menulis fungsi melakukan pengecekan margin, apabila *space* untuk penulisan *stego text* sudah menyentuh margin maka fungsi akan membuat halaman pdf baru seperti pada Gambar 4.18.

```
25  FOR line in lines
26      currentY <- currentY - leading
27      IF currentY <= marginbawah THEN
28          IF opsigeser == 3 THEN
29              FOR k=0 to k<counteropsitiga
30                  showText with arrayline_temp[k] in content
31                  newLineAtOffset with 0,-leading in content
32              counteropsitiga <-- 0
33          ELSE IF opsigeser == 2 THEN
34              FOR k=0 to k<counteropsidua
35                  showText with arrayline_temp[k] in content
36                  newLineAtOffset with 0,-leading in content
37              counteropsidua <-- 0
38          endText in content
39          content.close()
40          new_Page <-- new PDPage size A4
41          add this new_Page to doc
42          content <-- new PDPageContentStream in doc, new_Page
43          beginText in  content
44          setFont with pdfFont, size=11 in content
45          newLineAtOffset with startX, startY in content
46          currentY <-- startY
47          jmlbarisopsi <-- jmlbaris
48          jmlbaris <-- 0
49      jmlbaris <-- jmlbaris + 1
```

Gambar 4.18 Implementasi fungsi embed_stegotextpdf bagian kedua

Apabila *space* masih tersisa, fungsi akan melakukan penulisan *stego text* untuk tiap baris. Penulisan *stego text* dilakukan sesuai opsi geser yang dipilih. Dimana fungsi embed_stegotextpdf akan memanggil fungsi shiftopsisatu bila pilihan penggeseran baris yang dipilih adalah opsi baris genap. Kemungkinan lainnya, fungsi embed_stegotextpdf akan memanggil fungsi shiftopsidua bila pilihan penggeseran baris yang dipilih adalah opsi baris 1-5. Sementara untuk opsi baris 1-4-7, fungsi embed_stegotextpdf akan memanggil fungsi shiftopsitiga. Setelah penulisan selesai, fungsi akan menyimpan dokumen PDF *stego text* pada direktori yang sudah ditentukan oleh sistem. Rincian penulisan *stego text* dan

penyimpanan dokumen PDF *stego text* ditunjukkan oleh Gambar 4.19.

```
    IF opsigeser == 1 THEN
        cek_count <-- CALL shiftopsisatu with marginbawah, currentY,
        leading, contentStream, jmlbaris, jmlbarisopsi, listsize, line,
        countshift, a_binerpesan RETURNING cek_count
        countshift <-- countshift + cek_count
    ELSE IF opsigeser == 2 THEN
        cek_count_2 <-- CALL shiftopsidua with
        arrayline_temp,counteropsidua, marginbawah, currentY, leading,
        contentStream, jmlbaris, jmlbarisopsi, listsize, line, countshift,
        a_binerpesan RETURNING cek_count
        counteropsidua <-- counteropsidua + cek_count_2
        IF jmlbaris%5 == 0 THEN
            counteropsidua <-- 0
            countshift <-- countshift + 3
    ELSE IF opsigeser == 3 THEN
        cek_count_3 <-- CALL shiftopsitiga with arrayline_temp,
        counteropsitiga, marginbawah, currentY, leading, contentStream,
        jmlbaris, jmlbarisopsi, listsize, line, countshift, a_binerpesan
        RETURNING cek_count
        counteropsitiga <-- counteropsitiga + cek_count_3
        IF jmlbaris%3 == 1 and jmlbaris != 1 THEN
            counteropsitiga <-- 0
            countshift <-- countshift + 2
endText in content
close content
save doc to path
IF doc != null THEN
    close doc
RETURN true
ENDFUNCTION
```

Gambar 4.19 Implementasi fungsi embed_stegotextpdf bagian ketiga

Fungsi shiftopsisatu merupakan fungsi untuk menggeser baris pada *cover text* sesuai dengan opsi baris genap. Rincian fungsi shiftopsisatu seperti pada Gambar 4.20. Fungsi akan mengecek baris mana saja yang termasuk baris genap. Pengecekan pertama adalah pengecekan baris terakhir dari *cover text.* Baris terakhir tidak mengalami penggeseran, dan hanya akan melalui normalisasi rarangken sebelum ditulis dalam dokumen PDF. Baris-baris genap akan digeser posisinya dengan mengganti *Unicode* dari karakter Aksara Sunda pada baris. Sebelum mengalami penggeseran dengan memanggil fungsi shift, baris yang bergeser terlebih dahulu melalui normalisasi rarangken. Penggeseran posisi baris akan menyesuaikan isi dari a_binerpesan(*array of char*). Bilamana isi a_binerpesan bernilai 0,

baris akan digeser turun. Bila isi a_binerpesan bernilai 1 baris akan digeser naik. Sementara baris ganjil mengalami perlakuan yang sama dengan baris terakhir, yaitu melalui normalisasi rarangkén kemudian ditulis ke dalam dokumen PDF. Fungsi akan mengembalikan nilai cek_count (*integer*) yang digunakan oleh fungsi embed_stegotextpdf sebagai *counter* indeks dari a_binerpesan(*array of char*) yang berisi biner *secret message*.

```
1  FUNCTION shiftopsisatu(marginbawah, currentY, leading, content, jmlbaris,
   jmlbarisopsi, listsize, line, countshift, a_binerpesan):
2  linecetak <-- ""
3  hasilshift <-- ""
4  tempnormal <-- ""
5  cek_count <-- 0
6  IF jmlbaris + jmlbarisopsi == listsize THEN
7      linecetak <-- CALL normalisasi_rarangken with line RETURNING line_normal
8      showText with linecetak in content
9      newLineAtOffset with 0, -leading in content
10 ELSE IF jmlbaris % 2 == 0 and currentY - leading > marginbawah THEN
11     tempnormal <-- CALL normalisasi_rarangken with line RETURNING line_normal
12     hasilshift <-- CALL shift with a_binerpesan[countshift], tempnormal
       RETURNING baris
13     showText with hasilshift in content
14     newLineAtOffset with 0, -leading in content
15     cek_count <-- 1
16     RETURN cek_count;
17 ELSE
18     linecetak <-- CALL normalisasi_rarangken with line RETURNING line_normal
19     showText with linecetak in content
20     newLineAtOffset with 0, -leading in content
21 RETURN cek_count;
22 ENDFUNCTION
```

Gambar 4.20 Implementasi fungsi shiftopsisatu

Fungsi shiftopsidua merupakan fungsi untuk menggeser baris pada *cover text* sesuai dengan opsi baris 1-5. Rincian fungsi shiftopsidua seperti pada Gambar 4.21. Fungsi akan mengecek baris mana saja yang termasuk baris yang digeser. Baris yang digeser merupakan baris yang berada diantara dua baris tetap. Baris tetap akan melalui normalisasi rarangkén sebelum ditulis ke dokumen PDF. Sementara untuk baris yang bergeser, fungsi akan memanggil fungsi shift yang menggeser posisi baris setelah memanggil fungsi normalisasi_rarangken untuk menormalkan posisi rarangkén. Penggeseran posisi baris akan menyesuaikan isi dari a_binerpesan(*array of char*). Bilamana isi a_binerpesan bernilai 0, baris akan digeser turun. Bila isi a_binerpesan bernilai 1

baris akan digeser naik. Sementara baris terakhir akan melalui normalisasi rarangkén kemudian ditulis ke dalam dokumen PDF. Fungsi akan mengembalikan nilai cek_count (*integer*) yang digunakan oleh fungsi embed_stegotextpdf sebagai *counter* indeks dari a_binerpesan(*array of char*) yang berisi biner *secret message*.

```
1  FUNCTION
   shiftopsidua(arrayline_temp,counteropsidua,marginbawah,currentY,leading,content,
   jmlbaris,jmlbarisopsi,listsize,line,countshift,a_binerpesan):
2  linecetak,hasilshift,tempnormal <-- ""
3  cek_count <-- 0
4  IF jmlbaris%5 == 0 THEN
5      arrayline_temp[counteropsidua] <-- line
6      FOR j=0 to j<=counteropsidua=
7          IF j==0 or j==4 THEN
8              linecetak <-- CALL normalisasi_rarangken with arrayline_temp[j]
               RETURNING line_normal
9              showText with linecetak in content
10             newLineAtOffset with 0, -leading in content
11         ELSE
12             tempnormal <-- CALL normalisasi_rarangken with arrayline_temp[j]
               RETURNING line_normal
13             hasilshift <-- CALL shift with a_binerpesan[countshift],tempnormal
               RETURNING baris
14             countshift <-- countshift + 1
15             showText with hasilshift in content
16             newLineAtOffset with 0, -leading in content
17         counteropsidua <-- 0
18 ELSE IF jmlbaris + jmlbarisopsi == listsize THEN
19     FOR i=0; to i<counteropsidua
20         linecetak <-- CALL normalisasi_rarangken with arrayline_temp[i]
           RETURNING line_normal
21         showText with linecetak in content
22         newLineAtOffset with 0,-leading in content
23     linecetak <-- CALL normalisasi_rarangken with line RETURNING line_normal
24     showText with linecetak in content
25     newLineAtOffset with 0,-leading in content
26 ELSE IF jmlbaris%5 > 0 THEN
27     arrayline_temp[counteropsidua] <-- line
28     cek_count <-- 1
29     RETURN cek_count
30 RETURN cek_count
31 ENDFUNCTION
```

Gambar 4.21 Implementasi fungsi shiftopsidua

Fungsi shiftopsitiga merupakan fungsi untuk menggeser baris pada *cover text* sesuai dengan opsi baris 1-4-7. Rincian fungsi shiftopsitiga seperti pada Gambar 4.22 dan Gambar 4.23. Fungsi akan mengecek baris mana saja yang termasuk baris yang digeser. Baris yang digeser merupakan baris yang berada diantara dua baris tetap. Pada baris yang bergeser, fungsi akan memanggil fungsi shift yang berguna untuk menggeser posisi baris setelah memanggil fungsi normalisasi_rarangken untuk menormalkan posisi

rarangken. seperti pada Gambar 4.24. Penggeseran posisi baris akan menyesuaikan isi dari a_binerpesan(*array of char*). Bilamana isi a_binerpesan bernilai 0, baris akan digeser turun. Bila isi a_binerpesan bernilai 1 baris akan digeser naik.

```
1  FUNCTION
   shiftopsitiga(arrayline_temp,counteropsitiga,marginbawah,currentY,leading,content,
   jmlbaris,jmlbarisopsi,listsize,line,countshift,a_binerpesan):
2  linecetak , hasilshift, tempnormal <-- ""
3  cek_count <-- 0
4  IF jmlbaris == 1 THEN
5      linecetak <-- CALL normalisasi_rarangken with line RETURNING line_normal
6      showText with linecetak in content
7      newLineAtOffset with 0,-leading in content
8      counteropsitiga <-- 0
9  ELSE IF jmlbaris%3 == 1 THEN
10     arrayline_temp[counteropsitiga] <-- line
11     FOR j=0 to j<=counteropsitiga
12         IF j==2 THEN
13             linecetak <-- CALL normalisasi_rarangken with arrayline_temp[j]
               RETURNING line_normal
14             showText with linecetak in content
15             newLineAtOffset with 0, -leading in content
16         ELSE
17             tempnormal <-- CALL normalisasi_rarangken with arrayline_temp[j]
               RETURNING line_normal
18             hasilshift <-- CALL shift with a_binerpesan[countshift],tempnormal
               RETURNING baris
19             showText with hasilshift in content
20             countshift <-- countshift + 1
21             newLineAtOffset with 0, -leading in content
22     counteropsitiga <-- 0
```

Gambar 4.22 Implementasi fungsi shiftopsitiga bagian pertama

Baris tetap akan melalui normalisasi rarangken sebelum ditulis ke dokumen PDF. Sementara baris terakhir akan melalui normalisasi rarangken kemudian ditulis ke dalam dokumen PDF. Penulisan baris tetap dan baris geser sendiri dilakukan saat fungsi menemukan baris terakhir atau baris kelipatan 3 dari baris pertama. Sebelumnya, baris tetap akan disimpan dalam arrayline_temp (*array of string*). Fungsi shiftopsitiga akan mengembalikan nilai cek_count (*integer*) yang digunakan oleh fungsi embed_stegotextpdf sebagai *counter* indeks dari a_binerpesan(*array of char*) yang berisi biner *secret message*.

```
ELSE IF jmlbaris%3 == 1 THEN
    arrayline_temp[counteropsitiga] <-- line
    FOR j=0 to j<=counteropsitiga
        IF j==2 THEN
            linecetak <-- CALL normalisasi_rarangken with arrayline_temp[j]
            RETURNING line_normal
            showText with linecetak in content
            newLineAtOffset with 0, -leading in content
        ELSE
            tempnormal <-- CALL normalisasi_rarangken with arrayline_temp[j]
            RETURNING line_normal
            hasilshift <-- CALL shift with a_binerpesan[countshift],tempnormal
            RETURNING baris
            showText with hasilshift in content
            countshift <-- countshift + 1
            newLineAtOffset with 0, -leading in content
    counteropsitiga <-- 0
ELSE IF jmlbaris + jmlbarisopsi == listsize THEN
FOR i=0 to i<counteropsitiga
    linecetak <-- CALL normalisasi_rarangken arrayline_temp[i] RETURNING
    line_normal
    showText with linecetak in content
    newLineAtOffset with 0,-leading in content
linecetak <-- CALL normalisasi_rarangken with line RETURNING line_normal
showText with linecetak in content
newLineAtOffset with 0,-leading in content
ELSE IF jmlbaris%3 != 1 THEN
    arrayline_temp[counteropsitiga] <-- line
    cek_count <-- 1
    RETURN cek_count
RETURN cek_count
ENDFUNCTION
```

Gambar 4.23 Implementasi fungsi shiftopsitiga bagian kedua

Fungsi shiftopsisatu, fungsi shiftopsidua dan fungsi shiftopsitiga masing-masing memanggil fungsi shift dan fungsi normalisasi_rarangken di dalam fungsi itu sendiri. Fungsi shift menggeser posisi baris dengan menukar nilai Unicode dari karakter Aksara Sunda pada baris. Sementara fungsi normalisasi_rarangken menormalkan posisi rarangken dengan mengubah urutan karakter rarangken.

Fungsi normalisasi_rarangken memiliki parameter line (*string*) yang berisi seluruh karakter dalam satu baris. Fungsi akan mengubah line (*string*) ke dalam arraystring_line (*array of string*) untuk kemudian dilakukan pengecekan. Pada kondisi IF baris 12 fungsi melakukan normalisasi untuk rarangken panyuku yang berpasangan. Pada kondisi ELSE IF baris 32 fungsi normalisasi_rarangken memanggil fungsi cekrarangkensejajar untuk melakukan normalisasi rarangkén penghulu, paneuleung dan pamepet yang berpasangan serta rarangken paneling. Pada kondisi

ELSE IF baris 37 fungsi melakukan normalisasi untuk rarangken paneling. Rincian fungsi normalisasi_rarangken seperti pada Gambar 4.24.

```
1  FUNCTION normalisasi_rarangke(line):
2  line_normal <-- ""
3  esc_line <-- ""
4  esc_line_2 <-- ""
5  esc_line_3 <-- ""
6  arraystring_line <-- line.split("")
7  FOR i=0 to i<arraystring_line.length
8          esc_line <-- StringEscapeUtils.escapeJava(arraystring_line[i]);
9          kosong <-- ""
10         IF i+1 < arraystring_line.length THEN
11                 esc_line_2 <-- StringEscapeUtils.escapeJava(arraystring_line[i+1])
12                 IF esc_line_2 == "\\u1BA5" THEN
13                         temp <-- "\\u1BA5"
14                         IF esc_line == "\\u1BA1" THEN
15                                 temp <-- "\\u1BC8"
16                         ELSE IF esc_line == "\\u1BA2" THEN
17                                 temp <-- "\\u1BC6"
18                         ELSE IF esc_line == "\\u1BA3" THEN
19                                 temp <-- "\\u1BC7"
20                         IF temp == "\\u1BA5" THEN
21                                 esc_line <-- esc_line_2 + esc_line
22                                 increment i
23                         ELSE IF i+2 < arraystring_line.length THEN
24                                 esc_lain <-- StringEscapeUtils.escapeJava(arraystring_line[i+2])
25                                 IF CALL checkrarangken with esc_lain.replace("\\u","") RETURNING status is true THEN
26                                         kosong <-- kosong + esc_lain + temp
27                                         esc_line <-- kosong
28                                         i <-- i + 2
29                         ELSE
30                                 esc_line <-- temp
31                                 increment i
32         ELSE IF CALL cekrarangkensejajar with esc_line,esc_line_2 RETURNING status is true THEN
33             rarangken_terganti <-- CALL gantiunicoderarangken with esc_line, esc_line_2 RETURNING rarangken
34             IF rarangken_terganti != esc_line THEN
35                         increment i
36                 esc_line <-- rarangken_terganti
37         ELSE IF i+2 < arraystring_line.length THEN
38             esc_line_3 <-- StringEscapeUtils.escapeJava(arraystring_line[i+2])
39             IF esc_line_3 == "\\u1BA6" THEN
40                 escaped_line <-- kosong + esc_line_3 + esc_line + esc_line_2
41                 i <-- i + 2
42     ELSE
43                 esc_line <-- StringEscapeUtils.escapeJava(arraystring_line[i])
44         unescaped_line <-- StringEscapeUtils.unescapeJava(esc_line)
45         line_normal <-- line_normal + unescaped_line
46 RETURN line_normal
47 ENDFUNCTION
```

Gambar 4.24 Implementasi fungsi normalisasi_rarangken

Fungsi checkrarangken dipanggil oleh fungsi normalisasi_rarangken untuk menangani normalisasi pada rarangken panyuku. Rincian fungsi chekcrarangken seperti pada Gambar 4.25. Fungsi checkrarangken memiliki parameter rarangken(*string*). Parameter yang dimasukkan kemudian akan dicek termasuk rarangkén atau bukan dengan diubah ke dalam nilai hex_val (integer). Apabila nilai hex_val berada dalam rentang sesuai kondisi IF pada baris 7 maka rarangken(*string*) termasuk rarangken dan fungsi mengembalikan status “true”.

```
1  FUNCTION checkrarangken(rarangken):
2  hex_val <-- 0
3  stripped_unicode <-- rarangken.replace("\\u", "")
4  IF stripped_unicode.length() !=4 THEN
5      RETURN false
6  hex_val <-- Integer.parseInt(stripped_unicode) to HEX
7  IF (hex_val > 7072 && hex_val < 7083) or (hex_val > 7103
   && hex_val < 7113) or (hex_val > 7039 && hex_val < 7043)
   THEN
8      RETURN true
9  RETURN false
10 ENDFUNCTION
```

Gambar 4.25 Implementasi fungsi checkrarangken

Fungsi cekrarangkensejajar dipanggil oleh fungsi normalisasi_rarangken untuk menangani normalisasi rarangkén penghulu, paneuleung dan pamepet yang berpasangan serta rarangkén paneling. Rincian fungsi cekrarangkensejajar seperti pada Gambar 4.26. Fungsi cekrarangkensejajar memiliki parameter rarangken(*string*) dan rarangken_next (*string*). Parameter akan dicek seperti pada kondisi IF pada baris 2, jika kondisi terpenuhi maka fungsi akan mengembalikan status “true”.

```
1  FUNCTION cekrarangkensejajar(rarangken, rarangken_next):
2  IF rarangken == "\\u1BA4" or  rarangken == "\\u1BA8" or
   rarangken == "\\u1BA9" or rarangken_next == "\\u1BA6"
3      RETURN true
4  RETURN false
5  ENDFUNCTION
```

Gambar 4.26 Implementasi fungsi cekrarangkensejajar

Fungsi gantiunicoderarangken akan dipanggil oleh fungsi normalisasi_rarangken apabila fungsi cekrarangkensejajar yang dipanggil mengembalikan status “true”. Rincian fungsi

gantiunicoderarangken seperti pada Gambar 4.27. Pada kondisi IF baris 2 fungsi menukar nilai rarangkén penghulu. Pada kondisi ELSE IF baris 7 fungsi menukar nilai rarangkén pamepet. Pada kondisi ELSE IF baris 12 fungsi menukar nilai rarangkén paneuleung dan pada kondisi ELSE IF baris 17 fungsi menukar nilai rarangkén paneling.

```
1  FUNCTION gantiunicoderarangken(rarangken,rarangken_lain):
2  IF rarangken == "\\u1BA4" THEN
3      IF rarangken_lain == "\\u1B80" THEN
4          rarangken <-- "\\u1BC0"
5      ELSE IF rarangken_lain == "\\u1B81" THEN
6          rarangken <-- "\\u1BC3"
7  ELSE IF rarangken == "\\u1BA8" THEN
8      IF rarangken_lain == "\\u1B80" THEN
9          rarangken <-- "\\u1BC1"
10     ELSE IF rarangken_lain <-- "\\u1B81" THEN
11         rarangken = "\\u1BC4"
12 ELSE IF rarangken == "\\u1BA9" THEN
13     IF rarangken_lain == "\\u1B80" THEN
14         rarangken <-- "\\u1BC2"
15     ELSE IF rarangken_lain == "\\u1B81" THEN
16         rarangken = "\\u1BC5"
17 ELSE IF rarangken_lain == "\\u1BA6" THEN
18     rarangken <-- rarangken_lain + rarangken
19 RETURN rarangken;
20 ENDFUNCTION
```

Gambar 4.27 Implementasi fungsi gantiunicoderarangken

Rincian fungsi shift ditunjukkan pada Gambar 4.28. Fungsi shift memiliki parameter biner (*char*) dan barisdigeser (*string*). Jika biner bernilai 1, baris akan digeser naik sebaliknya baris akan digeser turun bila biner bernilai 0. Penggeseran dilakukan dengan diawali melakukan *split* pada barisdigeser (*string*) ke dalam arraystring_line (*array of string*). *String* yang ada pada arraystring_line akan diganti *Unicode*-nya. Fungsi shift memanggil fungsi gantiunicodenaik untuk penggantian *Unicode* pada baris yang posisinya bergeser naik, sementara fungsi shift akan memanggil fungsi gantiunicodeturun untuk penggantian *Unicode* pada baris yang posisinya bergeser turun.

```
FUNCTION shift(biner, barisdigeser):
baris <-- null
IF biner == '1' THEN
    barisnaik <-- ""
    arraystring_line <-- barisdigeser.split("")
    FOR i=0 to i<arraystring_line.length
        esc_char <--
        StringEscapeUtils.escapeJava(arraystring_line[i]);
        shifted_char <-- CALL gantiunicodenaik with
        esc_char) RETURNING unicode_naik
        unesc_char <--
        StringEscapeUtils.unescapeJava(shifted_char)
        barisnaik <-- barisnaik + unesc_char
    baris <-- barisnaik
ELSE IF biner == '0' THEN
    baristurun <-- ""
    arraystring_line <-- barisdigeser.split("")
    FOR i=0 to i<arraystring_line.length
        esc_char <--
        StringEscapeUtils.escapeJava(arraystring_line[i]);
        shifted_char <-- CALL gantiunicodeturun with
        esc_char RETURNING unicode_turun
        unesc_char <--
        StringEscapeUtils.unescapeJava(shifted_char);
        baristurun <-- baristurun + unesc_char
    baris <-- baristurun
RETURN baris
ENDFUNCTION
```

Gambar 4.28 Implementa0si fungsi shift

```
FUNCTION gantiunicodenaik(unicode_naik):
hex_val <-- 0
stripped_unicode <-- ""
stripped_unicode <-- unicode_naik.replace("\\u", "")
IF stripped_unicode.length != 4
    RETURN stripped_unicode
hex_val <-- parseInt(stripped_unicode) to HEX
hex_val <-- hex_val + 146
temp_unicode <-- cast hex_val to char
unicode_naik <-- temp_unicode to string
RETURN unicode_naik
ENDFUNCTION
```

Gambar 4.29 Implementasi fungsi gantiunicodenaik

Rincian fungsi gantiunicodenaik seperti pada Gambar 4.29. Fungsi memiliki parameter unicode_naik (*string*). Fungsi akan mengubah unicode_naik (*string*) yang sudah di *replace* menjadi stripped_unicode (*string*) ke dalam hex_val (*integer*). Nilai hex_val kemudian akan ditambahkan 146. Nilai hex_val yang telah ditambahkan diubah menjadi temp_unicode (*char*) yang kemudian

diubah kembali menjadi unicode_naik (*string*). Fungsi akan mengembalikan nilai unicode_naik (*string*).

Rincian fungsi gantiunicodeturun seperti pada Gambar 4.30. Fungsi memiliki parameter unicode_turun (*string*). Fungsi akan mengubah unicode_turun (*string*) yang sudah di *replace* menjadi stripped_unicode (*string*) ke dalam hex_val (*integer*). Nilai hex_val kemudian akan ditambahkan 73. Nilai hex_val yang telah ditambahkan diubah menjadi temp_unicode (*char*) yang kemudian diubah kembali menjadi unicode_turun (*string*). Fungsi akan mengembalikan nilai unicode_turun (*string*).

```
1  FUNCTION gantiunicodeturun(unicode_turun):
2  hex_val <-- 0
3  stripped_unicode <-- ""
4  stripped_unicode <-- unicode_turun.replace("\\u", "")
5  IF stripped_unicode.length != 4 THEN
6      RETURN stripped_unicode
7  hex_val <-- parseInt(stripped_unicode) to HEX
8  hex_val <-- hex_val + 73
9  temp_unicode <-- cast hex_val to char
10 unicode_turun <-- temp_unicode to string
11 RETURN unicode_turun
12 ENDFUNCTION
```

Gambar 4.30 Implementasi fungsi gantiunicodeturun

4.6. Implementasi Modul Ekstraksi

Pada subbab ini akan dijelaskan tentang implementasi dari modul ekstraksi. Proses yang terjadi pada modul ekstraksi terdiri dari pembacaan *stego text*, penghitungan jumlah baris geser, pencarian bit biner *secret message* dan ekstraksi *secret message.*

4.6.1. Implementasi Pembacaan *Stego Text*

Proses pembacaan *stego text* dimulai dengan menerima data masukan *stego text* dalam bentuk PDF. Masukan ini dibaca ketika fungsi bacapdf dipanggil. Rincian proses pembacaan *stego text* seperti ditunjukkan pada Gambar 4.31.

```
READ opsigeser, pathpdf
secret_message <-- CALL bacapdf with opsigeser, pathpdf
RETURNING secret_message
OUT secret_message
```

Gambar 4.31 Proses pembacaan *stego text*

```
FUNCTION bacapdf(opsigeser, path):
stegotext <-- null
lines <-- []
bitbiner <-- ""
secret_message <-- ""
jumlahbaris <-- 0
document <-- load PDDocument with File(path)
document.getClass
IF document.isEncrypted RETURNING status is false THEN
    stripper <-- new PDFTextStripperByArea
    setSortByPosition with true in stripper
    tStripper <-- new PDFTextStripper
    pdfFileInText <-- getText with document in tStripper
    lines <-- pdfFileInText.split("\\r?\\n")
    FOR line in lines
        increment jumlahbaris
    stegotext <-- pdfFileInText
baris_geser <-- CALL hitungbarisgeser with opsigeser,
lines RETURNING baris_geser
bitbiner <-- CALL biner_stegotext with
lines,opsigeser,baris_geser,jumlahbaris RETURNING bitbiner
secret_message <-- CALL ekstraksi_secretmessage with
bitbiner RETURNING secret_message
RETURN secret_message
ENDFUNCTION
```

Gambar 4.32 Implementasi fungsi bacapdf

Fungsi bacapdf memiliki parameter opsigeser (*integer*) dan path (*string*). Fungsi bacapdf terlebih dahulu membuka dokumen PDF sesuai dengan lokasi file yang ada pada parameter path (*string*). Isi *stego text* yang dibaca disimpan dalam variabel stegotext (*string*). Fungsi bacapdf kemudian memanggil fungsi hitungbarisgeser untuk menghitung jumlah baris geser. Setelah itu, fungsi bacapdf memanggil fungsi biner_stegotext untuk mendapatkan bit biner *secret message*. Terakhir, fungsi bacapdf memanggil fungsi ekstraksi_secretmessage untuk mengkonversi bit biner menjadi karakter yang merepresentasikan *secret message* yang dicari. Fungsi bacapdf mengembalikan nilai secret_message (*string*). Rincian fungsi bacapdf seperti ditunjukkan pada Gambar 4.32.

Fungsi hitungbarisgeser memiliki parameter opsigeser (*integer*) dan lines (*array of string*). Fungsi hitungbarisgeser mengembalikan nilai barisgeser (*integer*) yang berisi jumlah baris geser. Rincian fungsi hitungbarisgeser seperti pada Gambar 4.33.

```
1  FUNCTION hitungbarisgeser(opsigeser, lines):
2  baris_geser <-- 0
3  counter_baris <-- 0
4  IF opsigeser == 1 THEN
5      FOR i=0 to i<lines.length
6          increment counter_baris
7          IF counter_baris % 2 == 0 THEN
8              increment baris_geser
9          IF lines.length % 2 == 0 THEN
10             baris_geser <-- baris_geser - 1
11 ELSE IF opsigeser == 2
12     FOR i=0 to i<lines.length
13         increment counter_baris
14         IF counter_baris % 5 == 0
15             baris_geser <-- baris_geser + 3
16 ELSE IF opsigeser == 3
17     FOR i=0 to i<lines.length
18         increment counter_baris
19         IF counter_baris != 1 and counter_baris % 3 == 1
20             baris_geser <-- baris_geser + 2
21 RETURN baris_geser
22 ENDFUNCTION
```

Gambar 4.33 Implementasi fungsi hitungbarisgeser

Fungsi biner_stegotext memiliki parameter opsigeser (*integer*), stegotext (*array of string*), baris_geser (*integer*) dan jumlahbaris (*integer*). Fungsi biner_stegotext akan memanggil fungsi cekbarisgeser untuk menentukan apakah baris yang sedang diproses dalam perulangan termasuk baris geser atau bukan. Jika fungsi cekbarisgeser memberikan kembalian status “true” maka baris termasuk baris geser lalu fungsi biner_stegotext akan memanggil fungsi cek_unicodestego untuk mengetahui nilai Unicode dari karakter yang terdapat dalam baris. Dari pemanggilan fungsi cek_unicodestego akan didapatkan nilai biner “1” atau “0” yang kemudian disimpan dalam variabel bitbiner (*string*). Selama proses *looping*, apabila kondisi IF pada baris 9 terpenuhi, fungsi akan berhenti. Fungsi biner_stegotext mengembalikan nilai bitbiner (*string*) yang berisi bit biner dari *secret message*. Rincian fungsi biner_stegotext seperti pada Gambar 4.34.

```
1  FUNCTION biner_stegotext(stegotext, opsigeser,
   baris_geser, jumlahbaris):
2  bitbiner <--  ""
3  panjang_biner <-- 0
4  FOR i=0 to i<stegotext.length
5      IF CALL cekbarisgeser with i+1,opsigeser,jumlahbaris
       RETURNING status is true THEN
6          temp_biner <-- CALL cek_unicodestego with
           stegotext[i] RETURNING bitbiner
7          bitbiner <-- bitbiner + temp_biner
8          increment panjang_biner
9      IF panjang_biner == baris_geser THEN
10         break
11 RETURN bitbiner
12 ENDFUNCTION
```

Gambar 4.34 Implementasi fungsi biner_stegotext

Fungsi cekbarisgeser memiliki parameter jumlahbaris (*integer*), urutanbaris (*integer*) dan opsigeser (*integer*). Fungsi cekbarisgeser mengembalikan status "true" apabila kondisi IF pada baris 2 atau ELSE IF pada baris 5 atau ELSE IF pada baris 12 terpenuhi. Rincian fungsi cekbarisgeser seperti pada Gambar 4.35.

```
1  FUNCTION cekbarisgeser(urutanbaris, opsigeser, jumlahbaris):
2  IF opsigeser == 1 THEN
3      IF urutanbaris % 2 ==0 and urutanbaris != jumlahbaris
       THEN
4          RETURN true
5  ELSE IF opsigeser == 2 THEN
6      IF jumlahbaris % 5 == 0 THEN
7          IF urutanbaris % 5 > 1 THEN
8          RETURN true
9      ELSE
10         IF urutanbaris % 5 > 1 and (jumlahbaris -
           urutanbaris) > jumlahbaris % 5
11             RETURN true
12 ELSE IF opsigeser == 3 THEN
13     IF jumlahbaris % 3 == 1 THEN
14         IF urutanbaris % 3 != 1 THEN
15             RETURN true
16     ELSE
17         IF urutanbaris % 3 != 1 and (jumlahbaris -
           urutanbaris) > 1 THEN
18             RETURN true
19 RETURN false
20 ENDFUNCTION
```

Gambar 4.35 Implementasi fungsi cekbarisgeser

```
1  FUNCTION cek_unicodestego(baris):
2  bitbiner <-- ""
3  hex_val <-- 0
4  escaped_baris <-- ""
5  escaped_baris <-- StringEscapeUtils.escapeJava(baris)
6  array_baris <-- baris.split("")
7  FOR karakter in array_baris
8      esc_karakter <--
       StringEscapeUtils.escapeJava(karakter).replace("\\u",
       "")
9      IF esc_karakter.length == 4 THEN
10         hex_val <-- parseInt(esc_karakter) to HEX
11         IF hex_val > 7112 and hex_val < 7186 THEN
12             bitbiner <-- bitbiner + "0"
13             break
14         ELSE IF hex_val > 7185 and hex_val < 7259 THEN
15             bitbiner <-- bitbiner + "1"
16             break
17 RETURN bitbiner
18 ENDFUNCTION
```

Gambar 4.36 Implementasi fungsi cek_unicodestego

Rincian fungsi cek_unicodestego seperti pada Gambar 4.36. Fungsi cek_unicodestego parameter baris (*string*). Fungsi cek_unicodestego mengubah parameter baris yang dimasukkan menjadi hex_val (*integer*). Apabila pengecekan nilai hex_val memenuhi kondisi IF pada baris 11, fungsi akan menambahkan nilai "1" pada bitbiner (*string*) dan nilai "0" bila memenuhi kondisi IF pada baris 14. Fungsi akan mengembalikan nilai bitbiner (*string*).

```
1  FUNCTION ekstraksi_secretmessage(bitbiner):
2  secret_message <-- ""
3  FOR i=0 to i<bitbiner.length/8
4      charcode <--
       parseInt(bitbiner.substring(8*i,(i+1)*8),2) to HEX
5      temp_char <-- cast charcode to char
6      secret_message <-- secret_message + temp_char
7  RETURN secret_message
8  ENDFUNCTION
```

Gambar 4.37 Implementasi fungsi ekstraksi_secretmessage

Fungsi ekstraksi_secretmessage memiliki parameter bitbiner (*string*). Fungsi akan mengubah nilai bitner (*string*) ke dalam bentuk hex dalam charcode (*integer*) untuk kemudin diubah menjadi lagi menjadi temp_char (*char*). Nilai temp_char yang didapatkan akan ditambahkan ke secret_message (*string*). Fungsi akan mengembalikan nilai secret_message (*string*) yang berisi

secret message yang berhasil diekstraksi. Rincian fungsi ekstraksi_secretmessage ditunjukkan pada Gambar 4.37.

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada sistem yang dikembangkan. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsionalitas sistem, *recovery*, *fidelity*, *robustness*, *capacity ratio*. Pengujian fungsionalitas program melihat apakah fungsi berjalan sesuai dengan yang direncanakanan, *recovery* dilakukan dengan mengetahui *secret message* dapat diekstraksi kembali dari *stego text*, *fidelity* dilakukan dengan membandingkan tingkat kemiripan *cover text* dan *stego text*, *robustness* dilakukan dengan menguji tingkat ketahanan *stego text* dalam mempertahankan *secret message*, *capacity ratio* dilakukan dengan mengetahui kemampuan *cover text* dalam menyimpan bit *secret message*.

5.1. Lingkungan Pengujian Sistem

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas seperti yang tertera pada Tabel 5.1.

Tabel 5.1 Lingkungan Implementasi Sistem

Perangkat	Jenis	Spesifikasi
Perangkat Keras	Prosesor	Intel® Core™ i3-6006U CPU @ 2.00 GHz 2.00 GHz
	Memori (*RAM*)	4 GB 2000 MHz DDR4
Perangkat Lunak	Sistem Operasi	Windows 10 Home Single Language
	Bahasa Pemrograman	Java
	Perangkat Pengembang (*IDE)*	Netbeans 8.0.2
	Word Editor	Ms. Word 2016

5.2. Data Pengujian

Data pengujian sistem steganografi teks pada Aksara Sunda adalah teks Aksara Sunda dalam bentuk docx sebagai *cover text*. Ukuran *cover text* dalam bentuk PDF digunakan untuk uji coba *capacity ratio* dan *fidelity*. Hasil cetak *stego text* dalam bentuk PDF dan cover text dalam bentuk docx digunakan untuk uji coba *fidelity*. Uji coba *robustness* menggunakan dokumen *stego text* dalam bentuk PDF. Teks Aksara Sunda yang digunakan merupakan teks puisi dan sajak. Rincian setiap data uji coba ditunjukkan pada Tabel 5.2.

Tabel 5.2 Data Pengujian

No.	Nama Berkas	Ukuran docx (kb)	Ukuran pdf (kb)	Sumber
1	Anaking.docx	12,7	15,8	Internet [23]
2	Apan Urang Lembur Tea.docx	12,8	16,9	Internet [24]
3	Dandanggula Sugan.docx	12,6	16,5	Internet [25]
4	Hidup.docx	12,4	13,9	Internet [26]
5	Katineung.docx	12,3	15,0	Internet [27]
6	Potret.docx	12,5	15,6	Internet [28]
7	Sajak-sajak Godi Suwarna.docx	18,4	25,9	Internet [29]
8	Sekar Merdika.docx	12,5	16,1	Internet [30]
9	Sinareng Bulan Purnama.docx	12,5	16,0	Internet [31]
10	Solawat Daun.docx	12,8	17,3	Internet [32]
11	Sajak Muhsin 500 kb.docx	124,0	531,0	Internet [30] dengan modifikasi

12	Sajak Muhsin 1mb.docx	243,0	1044,0	Internet [30] dengan modifikasi
13	Sajak Muhsin 2mb.docx	473,0	2095,0	Internet [30] dengan modifikasi
14	Sajak Muhsin 3mb.docx	704,0	3128,0	Internet [30] dengan modifikasi

5.3. Skenario Pengujian

Pengujian dilakukan untuk mengetahui ketercapaian dan performa sistem steganografi teks pada aksara Sunda. Ketercapaian dapat dilihat dari beberapa aspek yaitu fungsionalitas, *recovery*, *fidelity*, *robustness*, dan *capacity ratio*. Setiap uji coba memiliki skenario untuk mencoba kemungkinan dan mengetahui nilai yang sesuai untuk diterapkan.

5.3.1. Skenario Pengujian Fungsionalitas

Pengujian fungsionalitas bertujuan untuk mengetahui fungsionalitas yang dapat berjalan pada sistem. Fungsionalitas mencakup proses *embedding* dan ekstraksi pada opsi geser yaitu baris genap, baris 1-5, dan baris 1-4-7. Skenario untuk uji coba fungsionalitas terdiri dari:

1. Proses *embedding* dan ekstraksi dengan masukan *secret message* “kami” menggunakan opsi geser baris genap pada data ke-7,
2. Proses *embedding* dan ekstraksi dengan masukan *secret message* “kami” menggunakan opsi geser baris 1-5 pada data ke-7,
3. Proses *embedding* dan ekstraksi dengan masukan *secret message* “kami” menggunakan opsi geser baris 1-4-7 pada data ke-7,

5.3.2. Skenario Pengujian Recovery

Pengujian *recovery* bertujuan untuk mengetahui *secret message* yang telah disisipkan pada *cover text* dapat diekstraksi kembali dengan tepat dari *stego text*. Skenario untuk uji coba *recovery* terdiri dari :

1. *Recovery secret message* "x" dari *stego text* hasil *embedding* menggunakan opsi geser baris genap pada data ke-1,
2. *Recovery secret message* "di" dari *stego text* hasil *embedding* menggunakan opsi geser baris 1-5 pada data ke-3,
3. *Recovery secret message* "ok" dari stego text hasil *embedding* menggunakan opsi geser baris 1-4-7 pada data ke-2,
4. *Recovery secret message* "9" dari stego text hasil *embedding* menggunakan opsi geser baris genap pada data ke-4,
5. *Recovery secret message* "=" dari stego text hasil *embedding* menggunakan opsi geser baris 1-5 pada data ke-6,
6. *Recovery secret message* "@" dari stego text hasil *embedding* menggunakan opsi geser baris 1-4-7 pada data ke-5,
7. *Recovery secret message* "test" dari stego text hasil *embedding* menggunakan opsi geser baris genap pada data ke-7,
8. *Recovery secret message* "sony" dari stego text hasil *embedding* menggunakan opsi geser baris 1-5 pada data ke-7,
9. *Recovery secret message* "cavs" dari stego text hasil *embedding* menggunakan opsi geser baris 1-4-7 pada data ke-7,
10. *Recovery secret message* "empat" dari stego text hasil *embedding* menggunakan opsi geser baris 1-4-7 pada data ke-7,

5.3.3. Skenario Pengujian Fidelity

Pengujian *fidelity* bertujuan untuk tingkat kemiripan antara *cover text* dan *stego text*. Berkas *cover text* yang dbandingkan adalah berkas *cover text* dalam bentuk PDF dan berkas *stego text* dalam bentuk PDF. Pada pengujian *fidelity* akan digunakan

pengujian secara subjektif yang melibatkan 35 responden dengan menggunakan *Mean Opinion Score* (*MOS*). Skenario untuk uji coba *fidelity* terdiri dari :

1. Perbandingan dokumen PDF *cover text* dan *stego text* hasil penyisipan *secret message* "0" dengan menggunakan opsi geser baris genap pada data ke-8,
2. Perbandingan dokumen PDF *cover text* dan *stego text* hasil penyisipan *secret message* "0" dengan menggunakan opsi geser baris 1-5 pada data ke-8,
3. Perbandingan dokumen PDF *cover text* dan *stego text* hasil penyisipan *secret message* "0" dengan menggunakan opsi geser baris 1-4-7 pada data ke-8,
4. Perbandingan dokumen PDF *cover text* dan *stego text* hasil penyisipan *secret message* "x" dengan menggunakan opsi geser baris genap pada data ke-9,
5. Perbandingan dokumen PDF *cover text* dan *stego text* hasil penyisipan *secret message* "y" dengan menggunakan opsi geser baris 1-5 pada data ke-9,
6. Perbandingan dokumen PDF *cover text* dan *stego text* hasil penyisipan *secret message* "21" dengan menggunakan opsi geser baris 1-4-7 pada data ke 9,

Pengujian fidelity dengan *Mean Opinion Score* (*MOS*) merupakan suatu metode pengujian dengan mengukur kualitas *stego text* berdasarkan deskripsi kualitatif dari apa terlihat oleh mata. MOS memberikan indikasi numerik kualitas *stego text* yang didapatkan setelah disisipi *secret message*. Pada pengujian ini, beberapa responden diminta memberikan nilai rentang 1-5 untuk membandingkan kemiripan *stego text* dan *cover text*. Nilai 1 melambangkan nilai terburuk sedangkan nilai 5 melambangkan nilai terbaik. Nilai keseluruhan pengujian *MOS* akan dihitung dari nilai rata-rata hasil penilaian responden. Tabel parameter penilaian *MOS* dapat dilihat pada Tabel 5.3.

Tabel 5.3 Parameter Penilaian MOS

Nilai	Hasil Perbandingan	Keterangan
5	Sangat Baik	*Excellent*
4	Baik	*Good*
3	Sedang	*Fair*
2	Buruk	*Poor*
1	Sangat Buruk	*Bad*

5.3.4. Skenario Pengujian Robustness

Pengujian *robustness* bertujuan untuk mengetahui tingkat ketahanan *stego text* dalam mempertahankan isi *secret message* yang dikandungnya. *Stego text* hasil ketiga opsi geser (genap, 1-5 dan 1-4-7) diuji untuk diekstraksi kembali. Skenario untuk uji coba *robustness* terdiri dari:

1. Ekstraksi *stego text* hasil penyisipan *secret message* “a” pada data ke-10 yang telah dicetak dengan *printer*,
2. Ekstraksi *stego text* hasil penyisipan *secret message* “x” pada data ke-10 dengan menggunakan opsi geser baris genap, *stego text* telah melalui pencetakan ulang lewat proses *fotocopy* 2 kali,
3. Ekstraksi *stego text* hasil penyisipan *secret message* “7” pada data ke-10 dengan menggunakan opsi geser baris genap yang telah melalui pencetakan ulang lewat proses *fotocopy*,

5.3.5. Skenario Pengujian *Capacity Ratio*

Pengujian *capacity ratio* bertujuan untuk mengetahui kemampuan *cover text* dalam menyimpan bit *secret message*. *Capacity ratio* dihitung dengan membagi total rarangkén (bit) dan ukuran *cover text* (kiloByte). Pada pengujian ini besaran hasil *capacity ratio* adalah bit/kiloByte. Parameter yang digunakan adalah opsi geser. Skenario terdiri dari:

A. *Capacity ratio* dengan pilihan baris genap pada opsi geser,
B. *Capacity ratio* dengan pilihan baris 1-5 pada opsi geser,

C. *Capacity ratio* dengan pilihan baris 1-4-7 pada opsi geser,

5.3.6. Skenario Pengujian Jumlah Baris Geser dan Pengaruh *Font Size*

Pengujian jumlah baris geser dilakukan untuk mengetahui jumlah baris geser sesuai opsi baris geser yang dipilih dengan *cover text* yang bervariasi ukurannya . Parameter yang diubah pada uji coba ini meliputi *cover text* dengan ukuran mulai dari di bawah 500 kilobyte, *cover text* berukuran 500 kilobyte, 1 megabyte, 2 megabyte, dan 3 megabyte. Pengujian juga dilakukan dengan mengubah *font size* dokumen *stego text*, dimana dilihat pengaruh ukuran dari *font* terhadap jumlah baris geser. Pada pengujian *font size* dilakukan pengujian dengan *font size* mulai dari 8 sampai 20 dengan menggunakan data *cover text* ke-12.

5.3.7. Skenario Pengujian Panjang *Secret Message*

Uji coba panjang *secret message* bertujuan untuk mengetahui batas karakter yang dapat menjadi masukan. Hasil yang akan ditinjau adalah jumlah baris geser sesuai pilihan opsi geser yang dapat digunakan untuk menyimpan bit final dari *secret message*. Dari hasil jumlah baris geser yang terhitung dari *cover text* akan dilihat batas jumlah karakter yang dapat dimasukkan pada *cover text.* Parameter yang diubah pada pengujian ini adalah panjang dari *secret message* dimulai dengan *secret message* sepanjang 100 karakter, 500 karakter, 1000 karakter, 2000 karakter.

5.3.8. Skenario Pengujian Waktu

Uji coba waktu dilakukan untuk mengetahui waktu yang diperlukan untuk melakukan proses *embedding* dan ekstraksi. Waktu yang dihitung hanya waktu pada proses utama *embedding* yaitu penyisipan bit biner *secret message* dengan pergeseran baris.

Parameter yang diubah pada uji coba ini adalah ukuran *cover text* dan opsi geser.

5.4. Hasil Pengujian

Hasil uji coba berisi tentang hasil uji coba dari skenario yang telah dilakukan. Hasil-hasil skenario uji coba dibandingkan untuk dilakukan analisis dan evaluasi.

5.4.1. Hasil Pengujian Fungsionalitas

Rangkuman mengenai hasil pengujian fungsionalitas dapat dilihat pada Tabel 5.4. Berdasarkan data pada tabel tersebut, semua skenario pengujian berhasil dan program berjalan dengan baik. Sehingga bisa ditarik kesimpulan bahwa fungsionalitas dari aplikasi telah dapat bekerja sesuai dengan yang diharapkan.

Tabel 5.4 Hasil Pengujian Fungsionalitas

No	Fungsionalitas	Skenario	Hasil
1	Konversi *secret message*	Skenario 1	Sukses
		Skenario 2	Sukses
		Skenario 3	Sukses
2	Penghitungan jumlah baris geser	Skenario 1	Sukses
		Skenario 2	Sukses
		Skenario 3	Sukses
3	Penyisipan *secret message*	Skenario 1	Sukses
		Skenario 2	Sukses
		Skenario 3	Sukses
4	Penyimpanan *stego text*	Skenario 1	Sukses
		Skenario 2	Sukses
		Skenario 3	Sukses
5	Ekstraksi *secret message*	Skenario 1	Sukses
		Skenario 2	Sukses
		Skenario 3	Sukses

5.4.1.1. Pengujian Fungsionalitas Skenario 1

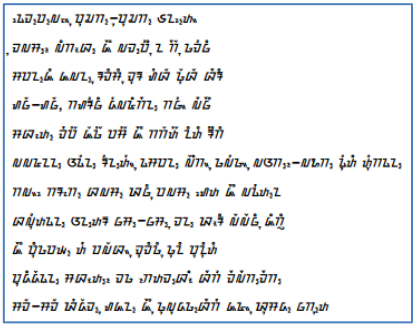
Secret message yang dimasukkan pada skenario 1 adalah "kami". Hasil konversi *secret message* menjadi bit biner seperti yang ditunjukkan pada Gambar 5.1 menghasilkan bit biner 01101011011000010110110101101001.

Gambar 5.1 Konversi *secret message* skenario 1

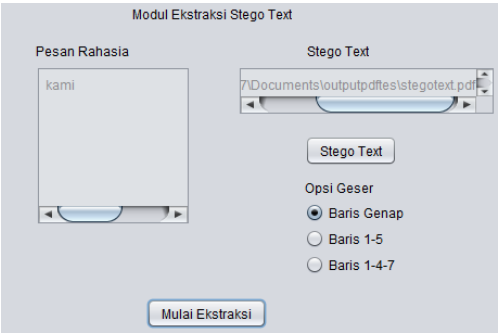
Hasil penghitungan jumlah baris dengan opsi geser baris genap seperti pada Gambar 5.2 menghasilkan 32 baris geser.

Gambar 5.2 Penghitungan jumlah baris geser skenario 1

Penyisipan *secret message* yang dilakukan kemudian menghasilkan *stego text* dalam bentuk PDF dengan nama file "stegotext.pdf" yang berisi seperti pada Gambar 5.3. Proses ekstraksi *secret message* yang dilakukan seperti pada Gambar 5.4 menghasilkan *secret message* "kami".



Gambar 5.3 Berkas *stego text* skenario 1



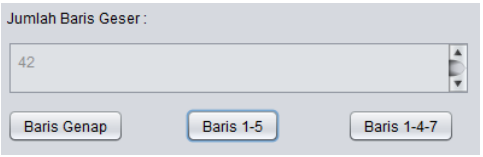
Gambar 5.4 Hasil ekstraksi skenario 1

5.4.1.2. Pengujian Fungsionalitas Skenario 2

Secret message yang dimasukkan pada skenario 2 adalah "kami". Hasil konversi *secret message* seperti pada Gambar 5.5 menghasilkan bit biner 01101011011000010110110101101001.

Gambar 5.5 Konversi *secret message* skenario 2

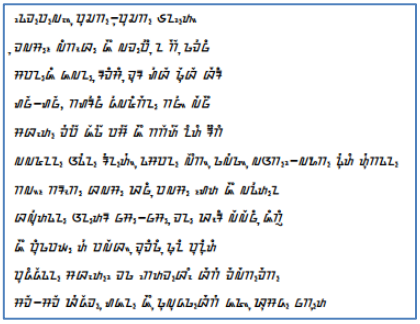
Hasil penghitungan jumlah baris dengan opsi geser baris 1-5 seperti pada Gambar 5.6 menghasilkan 42 baris geser.



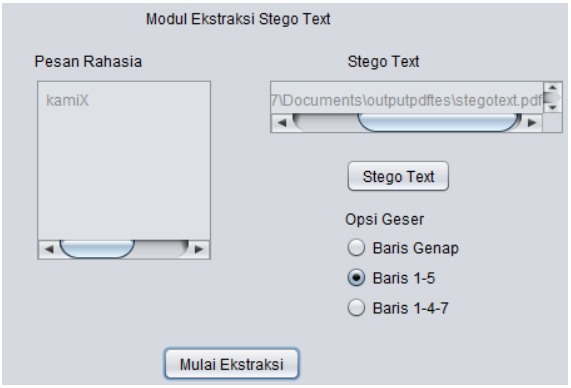
Gambar 5.6 Penghitungan jumlah baris geser skenario 2

Penyisipan *secret message* yang dilakukan kemudian menghasilkan *stego text* dalam bentuk PDF dengan nama file "stegotext.pdf" yang berisi seperti pada Gambar 5.7. Proses

ekstraksi *secret message* yang dilakukan seperti pada Gambar 5.8 menghasilkan *secret message* “kamiX”.



Gambar 5.7 Berkas *stego text* skenario 2



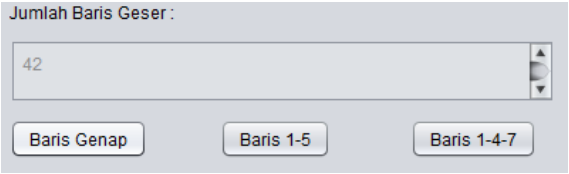
Gambar 5.8 Hasil ekstraksi skenario 2

5.4.1.3. Pengujian Fungsionalitas Skenario 3

Secret message yang dimasukkan pada skenario 3 adalah “kami”. Hasil konversi *secret message* seperti pada Gambar 5.9 menghasilkan bit biner 01101011011000010110110101101001.

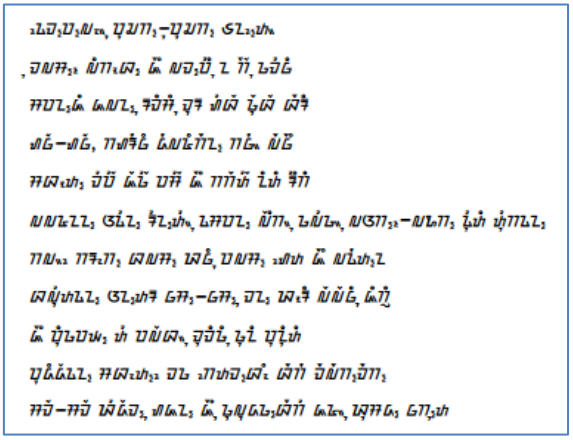
Gambar 5.9 Konversi *secret message* skenario 3

Hasil penghitungan jumlah baris dengan opsi geser baris 1-5 seperti pada Gambar 5.10 menghasilkan 42 baris geser.

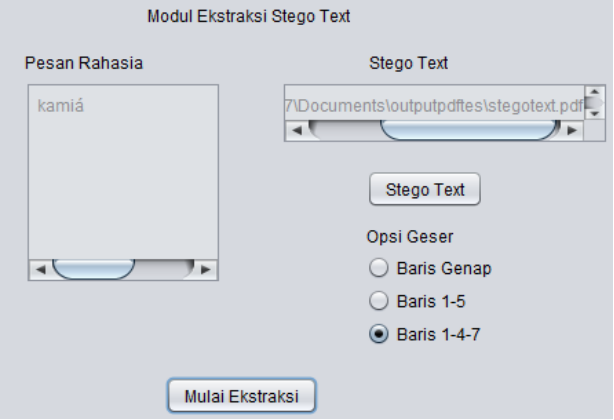


Gambar 5.10 Penghitungan jumlah baris geser skenario 3

Penyisipan *secret message* yang dilakukan kemudian menghasilkan *stego text* dalam bentuk PDF dengan nama file "stegotext.pdf" yang berisi seperti pada Gambar 5.11. Proses ekstraksi *secret message* yang dilakukan seperti pada Gambar 5.12 menghasilkan *secret message* "kamiá".



Gambar 5.11 Berkas *stego text* skenario 3



Gambar 5.12 Hasil ekstraksi skenario 3

5.4.2. Hasil Pengujian *Recovery*

Pengujian *recovery* dari sistem steganografi dilakukan dengan mengekstraksi *stego text* untuk mendapatkan kembali *secret message* yang sebelumnya disisipkan pada *cover text*. Indikasi sistem steganografi yang baik akan membuat pesan yang disisipkan dapat diambil kembali dengan utuh. Tingkat keutuhan *secret message* dilihat dari *secret message* hasil proses ekstraksi yang dibandingkan dengan *secret message* final.

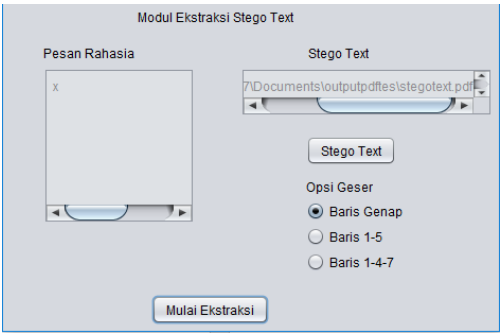
Secret message final mungkin saja berbeda dengan *secret message* awal yang disisipkan dikarenakan sistem akan menambahkan bit biner acak apabila jumlah baris geser pada *cover text* lebih banyak dari panjang bit biner hasil konversi *secret message*. Rangkuman pengujian *recovery* dapat dilihat pada Tabel 5.5. Ukuran *cover text* yang digunakan untuk pengujian bervariasi, dimana skenario 1 sampai 6 menggunakan *cover text* .docx berukuran 12 kB sedangkan pengujian skenario 7 sampai 10 menggunakan *cover text* .docx berukuran 18 kB.

Tabel 5.5 Rangkuman Hasil Pengujian *Recovery*

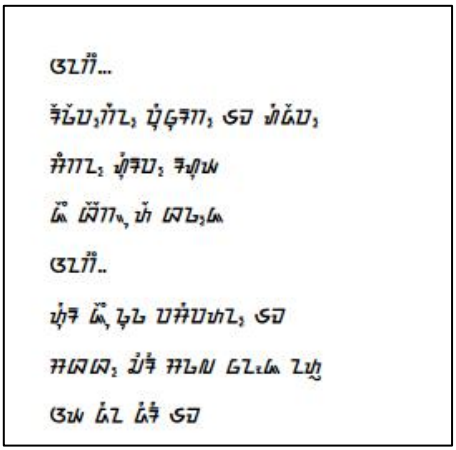
Skenario	*Secret Message* Yang Disipkan	*Secret Message* Final	*Secret Message* Hasil Ekstraksi
Skenario 1	x	x	x
Skenario 2	di	di	di
Skenario 3	ok	ok	ok
Skenario 4	9	9	9
Skenario 5	=	=	=
Skenario 6	@	@	@
Skenario 7	test	test	test
Skenario 8	sony	sonyD	sonyD
Skenario 9	cavs	cavs«	cavs«
Skenario 10	empat	Empat	empat

5.4.2.1. Pengujian *Recovery* Skenario 1

Dari hasil pengujian *recovery* scenario 1, proses ekstraksi pada *stego text* menghasilkan *secret message* "x" seperti pada Gambar 5.13. *Secret message* yang didapatkan dari ekstraksi *stego text* sendiri sama dengan *secret message* final yang disisipkan pada *cover text.* Berkas *stego text* seperti terlihat pada Gambar 5.14.



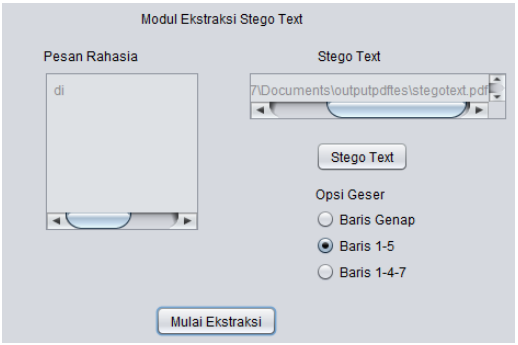
Gambar 5.13 Hasil ekstraksi pada skenario 1



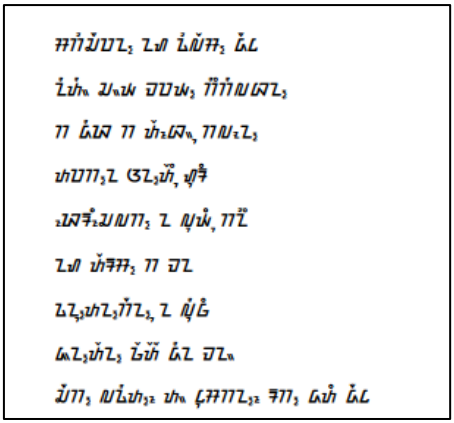
Gambar 5.14 Cuplikan berkas *stego text* skenario 1

5.4.2.2. Pengujian *Recovery* Skenario 2

Pada pengujian skenario 2, *secret message* yang disisipkan adalah “di”. *Stego text* skenario 2 seperti terlihat pada Gambar 5.16. Hasil pengujian skenario 2, *secret message* yang didapatkan kembali adalah “di” seperti pada Gambar 5.15 yang sama dengan *secret message* final yang disisipkan.



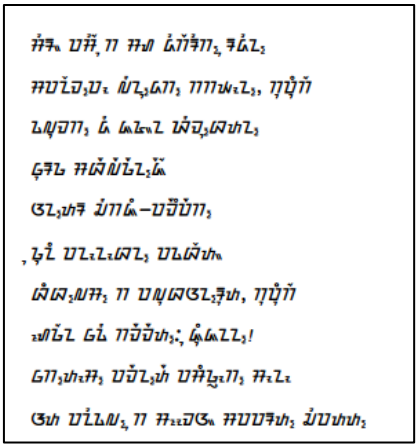
Gambar 5.15 Hasil ekstraksi skenario 2



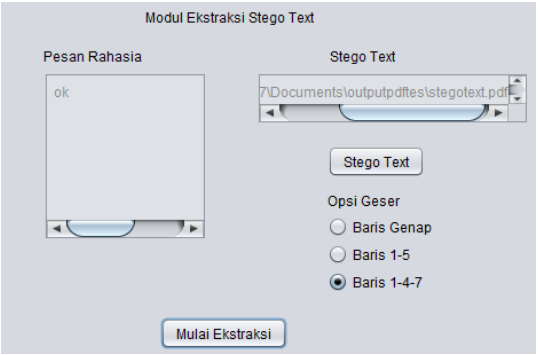
Gambar 5.16 Cuplikan berkas *stego text* skenario 2

5.4.2.3. Pengujian *Recovery* Skenario 3

Pada pengujian skenario 3, *secret message* final yang disisipkan adalah “ok”. Berksa *stego text* skenario 3 seperti terlihat pada Gambar 5.17. *Secret message* yang didapatkan kembali adalah “ok” seperti pada Gambar 5.18 yang sama dengan *secret message* final yang disisipkan oleh sistem.



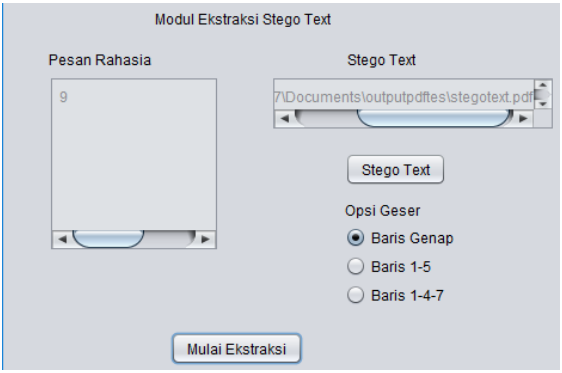
Gambar 5.17 Cuplikan *stego text* skenario 3



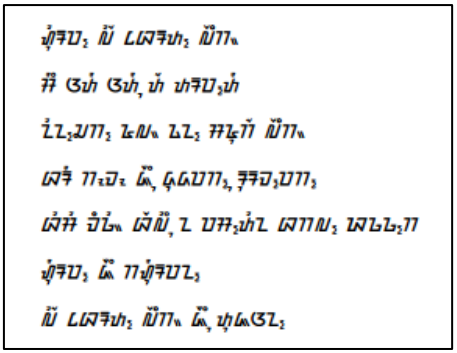
Gambar 5.18 Hasil ekstraksi skenario 3

5.4.2.4. Pengujian *Recovery* Skenario 4

Secret message final yang dimasukkan pada skenario 4 adalah "9" yang sama dengan *secret message* awal yang disisipkan pada *cover text*. Dari hasil pengujian, ekstraksi *stego text* menghasilkan *secret message* "9" seperti pada Gambar 5.19. Berkas dari *stego text* pada skenario 4 seperti terlihat pada Gambar 5.20.



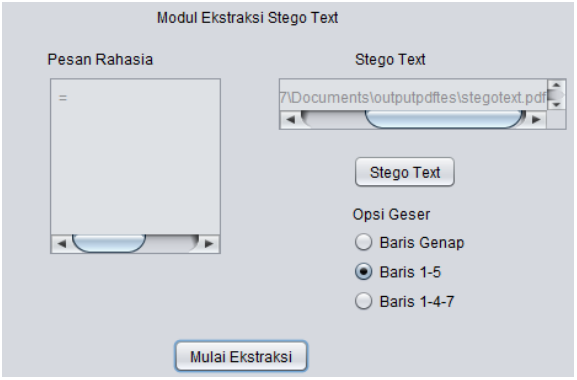
Gambar 5.19 Hasil ekstraksi skenario 4



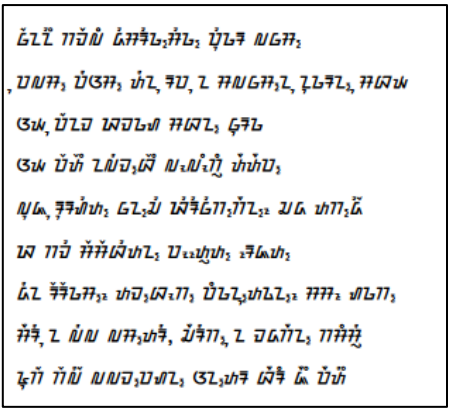
Gambar 5.20 Cuplikan *stego text* skenario 4

5.4.2.5. Pengujian *Recovery* Skenario 5

Pada skenario 5 *secret message* awal dan final adalah "=". Dimana hasil pengujian menunjukkan bahwa ekstraksi *stego text* menghasilkan *secret message* "=" seperti pada Gambar 5.21 yang sama dengan *secret messafe* final yang disisipkan oleh sistem. Berkas dari *stego text* pada skenario 5 seperti terlihat pada Gambar 5.22.



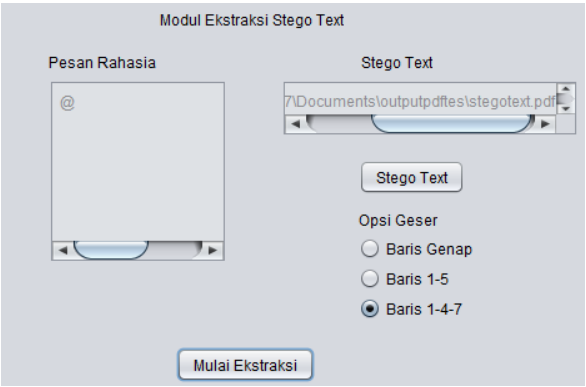
Gambar 5.21 Hasil ekstraksi skenario 5



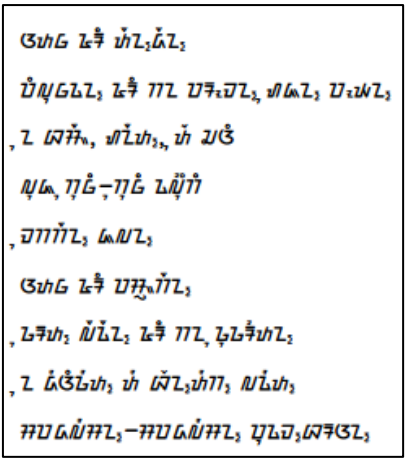
Gambar 5.22 Cuplikan *stego text* skenario 5

5.4.2.6. Pengujian *Recovery* Skenario 6

Pada skenario 6, *secret message* final yang disisipkan oleh sistem pada *cover text* adalah "@". Dari hasil pengujian terlihat bahwa proses ekstraksi *stego text* menghasilkan *secret message* "@" seperti pada Gambar 5.23 yang sesuai dengan *secret message* final. Berkas dari *stego text* skenario 6 seperti terlihat pada Gambar 5.24.



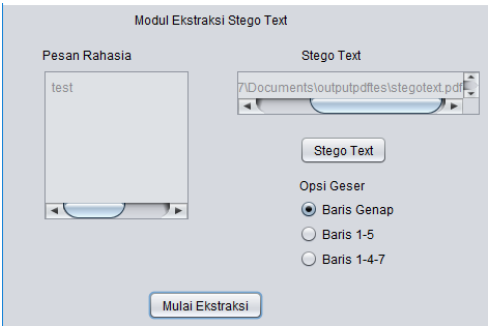
Gambar 5.23 Hasil ekstraksi skenario 6



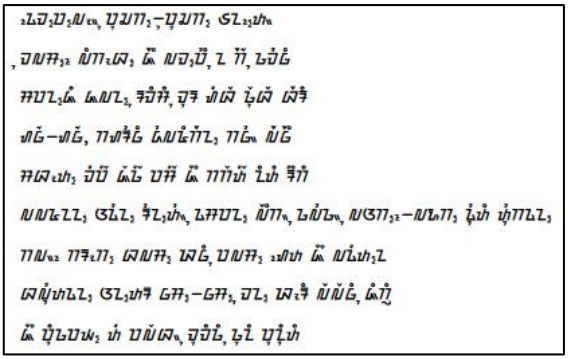
Gambar 5.24 Cuplikan *stego text* skenario 6

5.4.2.7. Pengujian *Recovery* Skenario 7

Pada skenario 7, *secret message* final yang disisipkan oleh sistem pada *cover text* adalah “test”. Dari hasil pengujian terlihat bahwa proses ekstraksi *stego text* menghasilkan *secret message* “test” yang sesuai dengan *secret message* final seperti ditunjukkan pada Gambar 5.25. Berkas dari *stego text* skenario 7 seperti terlihat pada Gambar 5.26.



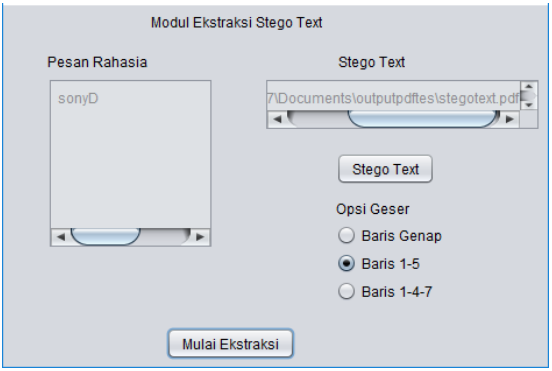
Gambar 5.25 Hasil ekstraksi skenario 7



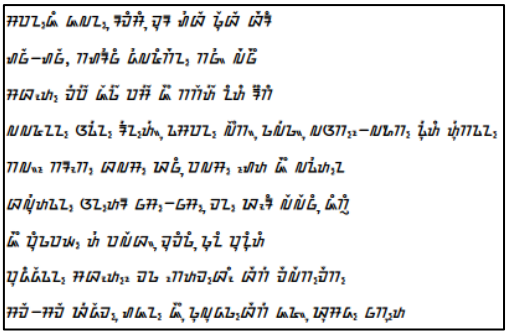
Gambar 5.26 Cuplikan *stego text* skenario 7

5.4.2.8. Pengujian *Recovery* Skenario 8

Pada skenario 8, *secret message* final yang disisipkan oleh sistem pada *cover text* adalah “sonyD”. Dari hasil pengujian terlihat bahwa proses ekstraksi *stego text* menghasilkan *secret message* “sonyD” yang sesuai dengan *secret message* final seperti ditunjukkan pada Gambar 5.27. Berkas dari *stego text* skenario 8 seperti terlihat pada Gambar 5.28.



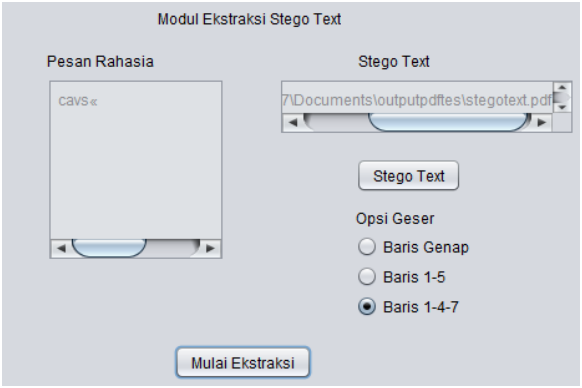
Gambar 5.27 Hasil eksktraksi skenario 8



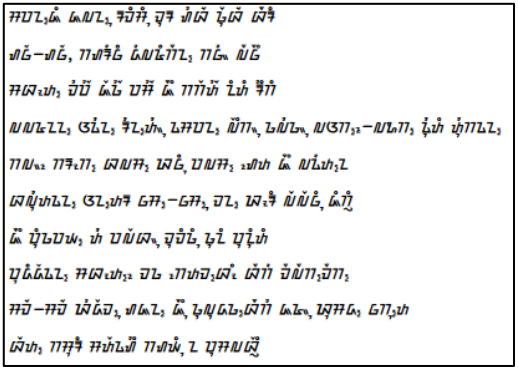
Gambar 5.28 Cuplikan *stego text* skenario 8

5.4.2.9. Pengujian *Recovery* Skenario 9

Pada skenario 9, *secret message* final yang disisipkan oleh sistem pada *cover text* adalah “cavs«”. Dari hasil pengujian terlihat bahwa proses ekstraksi *stego text* menghasilkan *secret message* “cavs«” yang sesuai dengan *secret message* final seperti ditunjukkan pada Gambar 5.29. Berkas dari *stego text* skenario 9 seperti terlihat pada Gambar 5.30.



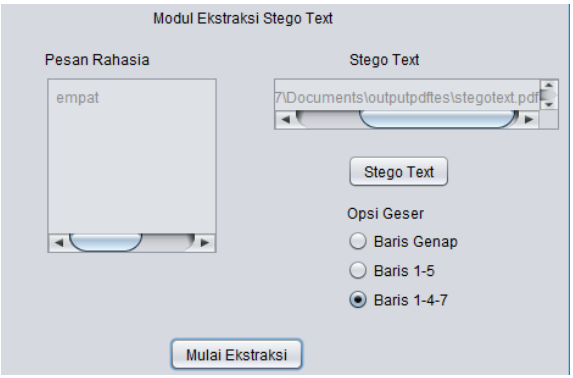
Gambar 5.29 Hasil ekstraksi skenario 9



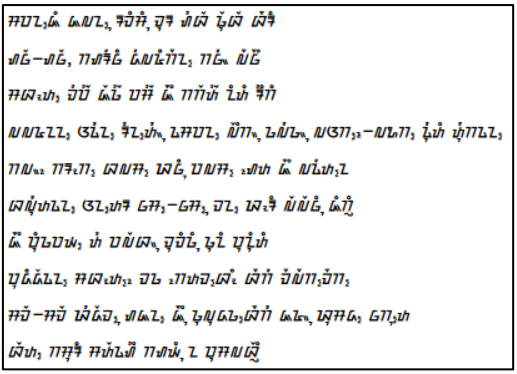
Gambar 5.30 Cuplikan berkas *stego text* skenario 9

5.4.2.10.Pengujian *Recovery* Skenario 10

Pada skenario 10, *secret message* final yang disisipkan oleh sistem pada *cover text* adalah “empat”. Dari hasil pengujian terlihat bahwa proses ekstraksi *stego text* menghasilkan *secret message* “empat” yang sesuai dengan *secret message* final seperti ditunjukkan pada Gambar 5.29. Berkas dari *stego text* skenario 10 seperti terlihat pada Gambar 5.30.



Gambar 5.31 Hasil ekstraksi skenario 10



Gambar 5.32 Cuplikan berkas *stego text* skenario 10

5.4.3. Hasil Pengujian *Fidelity*

Hasil Pengujian *fidelity* melaui metode *MOS* terangkum dalam Tabel 5.6. Dari hasil pengujian *fidelity* yang dilakukan, dapat diketahui bahwa tingkat kemiripan *cover text* dan *stego text* adalah sedang dengan nilai 3 pada pengujian *MOS*.

Tabel 5.6 Nilai *MOS* Hasil Pengujian

Skenario	Nilai *MOS*
Skenario 1	3
Skenario 2	3
Skenario 3	3
Skenario 4	3
Skenario 5	3
Skenario 6	3

Pengujian *fidelity* juga dilakukan dengan membandingkan ukuran (kilo/Byte) dari berkas PDF *cover text* dan *stego text*. Hasil perbandingan dapat dilihat pada Tabel 5.7 dimana rata-rata persentase perubahan ukuran mencapai 46,1%.

Tabel 5.7 Perubahan Ukuran Teks *Stego Text*

Skenario	Ukuran (kB)			Persentase Perubahan	
	PDF *cover text*	PDF *stego text*	Selisih	Per Skenario	Rata-rata
Skenario 1	16,1	8,58	7,52	46,7%	46,1%
Skenario 2	16,1	8,71	7,39	45,9%	
Skenario 3	16,1	8,55	7,55	46,8%	
Skenario 4	16,1	8,63	7,47	46,4%	
Skenario 5	16,1	8,73	7,37	45,8%	
Skenario 6	17,3	9,51	7,79	45,0%	

5.4.4. Hasil Pengujian *Robustness*

Rangkuman hasil pengujian *robustness* seperti terlihat pada Tabel 5.8. Hasil pengujian menunjukkan bahwa *stego text* mampu bertahan dari steganalisis berupa proses *fotocopy*. Dari hasil pengujian, *secret message* dapat diekstraksi kembali secara utuh meski telah melalui 2 kali proses *fotocopy*.

Tabel 5.8 Hasil Pengujian Tingkat Ketahanan

Opsi Geser	Skenario	Bit Biner *Secret Message* yang Disisipkan	Bit Biner *Secret Message* yang Diekstraksi
Baris Genap	Skenario 1	01100001	01100001
	Skenario 2	01111000	01111000
	Skenario 3	00110111	00110111
Baris 1-5	Skenario 1	011000010	011000010
	Skenario 2	011110000	011110000
	Skenario 3	001101111	001101111
Baris 1-4-7	Skenario 1	0110000110	0110000110
	Skenario 2	0111100010	0111100010
	Skenario 3	0011011100	0011011100

Proses ekstraksi pada uji *robustness* dilakukan dengan ekstraksi manual tanpa menggunakan sistem yang dibangun. Hal ini dapat dilakukan karena *cover text* dari dokumen teks memiliki jarak baris yang sama, oleh karena itu perbandingan jarak antar baris dapat digunakan untuk mengekstraksi kembali *stego text* hasil sistem, baik yang dicetak dengan *printer* maupun telah melewati pencetakan ulang dengan mesin *fotocopy*.

Gambar 5.33 Proses Ekstraksi *Stego Text* Uji *Robustness*

Proses ekstraksi kembali dengan mengukur jarak antar baris dilakukan seperti pada Gambar 5.33 dimana terdapat selisih jarak 0,05 mm pada baris yang bergeser jika jarak baris yang bergeser dibandingkan dengan baris yang menjadi *control groups*.

5.4.5. Hasil Pengujian *Capacity Ratio*

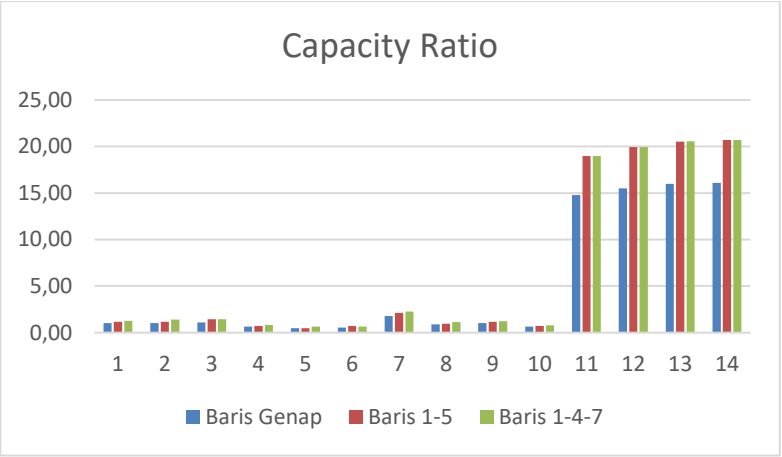
Pengujian *capacity ratio* dihitung dengan membagi total jumlah baris geser dengan ukuran *cover text*. Hasil penghitungan ditunjukan pada Tabel 5.9. Besaran hasil *capacity ratio* dihitung dalam bit/kiloByte.

Tabel 5.9 Hasil Pengujian *Capacity Ratio*

Data	Jumlah baris geser (bit)			Ukuran Berkas .docx *cover text* (kB)	Capacity Ratio (bit/kB)		
	Opsi Baris genap	Opsi baris 1-5	Opsi baris 1-4-7		Opsi Baris genap	Opsi baris 1-5	Opsi baris 1-4-7
5	6	6	8	12,3	0,49	0,49	0,65
4	8	9	10	12,4	0,65	0,73	0,81
6	7	9	8	12,5	0,56	0,72	0,64
8	11	12	14	12,5	0,88	0,96	1,12
10	8	9	10	12,5	0,64	0,72	0,8
3	14	18	18	12,6	1,11	1,43	1,43
1	13	15	16	12,7	1,02	1,18	1,26
2	13	15	18	12,8	1,02	1,17	1,41
9	13	15	16	12,8	1,02	1,17	1,25
7	32	42	42	18,4	1,74	2,28	2,28
11	1817	2334	2336	123	14,77	18,98	18,99
12	3771	4848	4848	243	15,52	19,95	19,95
13	7543	9696	9698	472	15,98	20,54	20,55
14	11314	14547	14546	703	16,09	20,69	20,69
				Rata-rata	5,11	6,50	6,56
				Rata-rata total			6,06

Hasil pengujian yang dilakukan menunjukkan besaran nilai *capacity ratio* pada tiap opsi geser. Opsi geser baris 1-4-7 memiliki *capacity ratio* paling tinggi mencapai 6,56 (bit/KB) namun tidak berselisih jauh dengan 2 opsi lain, dimana *capacity ratio* opsi geser baris 1-5 mencapai 6,50 (bit/KB) sedangkan opsi geser baris genap memiliki nilai *capacity ratio* terendah, yaitu 5,11 (bit/KB). Sedangkan nilai rata-rata *capacity ratio* dari sistem adalah 6,06 (bit/KB)

Nilai *capacity ratio* tertinggi ada pada data ke-14 yang memiliki *capacity ratio* mencapai 20,69 (bit/KB) untuk opsi geser baris 1-4-7 dan baris 1-5 serta 16,09 (bit/KB) untuk opsi geser baris genap. Grafik *capacity ratio* dapat dilihat pada Gambar 5.34.



Gambar 5.34 Grafik *capacity ratio*

5.4.6. Hasil Pengujian Jumlah Baris Geser dan Pengaruh *Font Size*

Hasil pengujian jumlah baris geser dapat dilihat pada

Tabel 5.10 dimana pada data *cover text* ke-14 dengan ukuran 3 megabyte, semua opsi geser menghasilkan jumlah baris geser lebih dari 10000 baris. Opsi geser baris 1-4-7 selalu menghasilkan jumlah baris geser terbanyak kecuali pada data *cover text* ke-14 dimana opsi geser baris 1-4-7 menghasilkan 14546 baris geser, sementara baris 1-5 menghasilkan baris geser hingga 14547. Variasi *cover text* di bawah 500 kilobyte menghasilkan persentase perubahan ukuran berkas PDF *stego text* sebesar 45,67%. Sedangkan *cover text* berukuran 500 kilobyte, 1 megabyte, 2 megabyte, dan 3 megabyte menghasilkan stego text yang mempunyai perubahan ukuran berkas PDF *stego text* 71,11%.

Tabel 5.10 Hasil Uji Coba Jumlah Baris Geser

Data	Ukuran Berkas PDF (kB)		Persentase Perubahan	Jumlah Baris		
	Cover Text	*Stego Text*		Opsi Baris Genap	Opsi Baris 1-5	Opsi Baris 1-4-7
4	13,9	7,58	45,47%	8	9	10
5	15	7,88	47,47%	6	6	8
6	15,6	8,72	44,10%	7	9	8
1	15,8	8,86	43,92%	13	15	16
10	16	8,93	44,19%	8	9	10
8	16,1	8,66	46,21%	11	12	14
3	16,5	8,91	46,00%	14	18	18
2	16,9	9,28	45,09%	13	15	18
9	17,3	9,55	44,80%	13	15	16
7	25,9	13,1	49,42%	32	42	42
11	531	152	71,37%	1817	2334	2336
12	1044	305	70,79%	3771	4848	4848
13	2095	604	71,17%	7543	9696	9698
14	3128	904	71,10%	11314	14547	14546

Pengujian ukuran *font* dilakukan pada data *cover text* ke-12. Hasil pengujian pengaruh dari ukuran *font* dapat dilihat pada Tabel 5.11 dimana ukuran *font* tidak berbanding lurus dengan jumlah baris geser. Ukuran font yang semakin besar tidak langsung membuat jumlah baris geser semakin kecil, hal ini juga dipengaruhi oleh margin pada dokumen yang berpengaruh pada jumlah maksimal baris dalam satu halaman pada dokumen. Meski tidak langsung berbanding lurus, namun hasil pengujian menunjukkan ukuran 8 pada *font* Sundanese Unicode 2013 menghasilkan jumlah baris geser terbanyak untuk semua opsi baris geser. Sedangkan ukuran 20 pada *font* Sundanese Unicode 2013 menghasilkan jumlah baris geser yang paling sedikit.

Tabel 5.11 Hasil Uji *Font Size*

Font Size	Jumlah Baris Geser		
	Opsi Baris Genap	Opsi Baris 1-5	Opsi Baris 1-4-7
8	3942	4731	5124
9	3931	4587	5242
10	3918	4410	4898
11	3771	4848	4848
12	3891	4488	4788
13	3879	4848	5172
14	3865	4215	4918
15	3673	4407	5142
16	3636	4848	4848
17	3828	3828	5104
18	3592	4041	4490
19	3803	4278	4754
20	3535	4545	5050

5.4.7. Hasil Pengujian Panjang *Secret Message*

Pada Tabel 5.12 ditunjukkan kemampuan dari tiap data yang digunakan sebagai *cover text* untuk menampung panjang karakter yang dimasukkan. Tanda “X” diberikan jika data tersebut tidak dapat menampung panjang karakter yang dimasukkan. Tanda “V” diberikan jika data tersebut dapat menampung panjang karakter.

Dari pengujian yang dilakukan hanya data ke-11, 12, 13 dan 14 yang dapat menampung masukan sepanjang 100 karakter. Sementara data ke-12, 13 dan 14 dapat menampung masukan dengan panjang hingga 500 karakter. Dari kesemua data yang diuji, tidak ada yang mampu menampung masukan dengan panjang 2000 karakter. Data ke-11, 12, 13 dan 14 dapat menampung masukan lumayan banyak karena memiliki jumlah baris geser yang lebih banyak dibanding data lainnya. Pada sistem steganografi, 1

karakter direpresentasikan sepanjang 8 bit biner, maka untuk 100 karakter direpresentasikan dalam 800 bit biner. Data ke-13 sendiri dapat menampung hingga 1000 karakter apabila menggunakan opsi geser baris 1-5 atau opsi geser baris 1-4-7.

Tabel 5.12 Hasil Uji Coba Panjang *Secret Message*

Data	Jumlah Baris Geser			Menampung *Secret Message* (karakter)			
	Baris Genap	Baris 1-5	Baris 1-4-7	100	500	1000	2000
1	13	15	16	X	X	X	X
2	13	15	18	X	X	X	X
3	14	18	18	X	X	X	X
4	8	9	10	X	X	X	X
5	6	6	8	X	X	X	X
6	7	9	8	X	X	X	X
7	32	42	42	X	X	X	X
8	11	12	14	X	X	X	X
9	13	15	16	X	X	X	X
10	8	9	10	X	X	X	X
11	1817	2334	2336	V	X	X	X
12	3771	4848	4848	V	V	X	X
13	7543	9696	9698	V	V	V	X
14	11314	14547	14546	V	V	V	X

5.4.8. Hasil Pengujian Waktu

Hasil pengujian waktu proses *embbeding* dapat dilihat pada Tabel 5.13. Tanda "-" menunjukkan jumlah baris pada *cover text* kurang dan tidak bisa disisipi minimal 1 karakter seperti pada data *cover text* ke-5 dan 6. Hasil pengujian menunjukkan sistem membutuhkan waktu rata-rata 111,51 milisekon untuk mengolah data *cover text* ke-2 dengan ukuran 16,9 kB. Sedangkan sistem membutuhkan waktu rata-rata hingga 40110,74 milisekon untuk

mengolah *cover text* berukuran 1010 kB. Penggunaan opsi genap memiliki waktu *running* tercepat pada data ke-1,3,4,7, 11,12, 13 dan 14. Opsi geser baris 1-5 tercepat pada data-2 dan 8 sementara opsi baris 1-4-7 tercepat saat memproses data ke-9 dan ke-10.

Tabel 5.13 Hasil Uji Coba Waktu *Embedding*

Data	Ukuran PDF *Cover Text* (kB)	Waktu Rata-rata *Embedding* (ms)			Waktu Rata-rata Total (ms)
		Opsi Baris Genap	Opsi Baris 1-5	Opsi Baris 1-4-7	
4	13,9	20,88	23,56	35,07	26,5
5	15	-	-	20,8	20,8
6	15,6	-	32,9	37,09	35
1	15,8	33,49	33,72	36,16	34,46
10	16	39,29	39,78	31,3	36,79
8	16,1	39,53	24,01	30,42	31,32
3	16,5	40,87	49,27	43	44,38
2	16,9	48,79	32,26	49,69	43,58
9	17,3	46,55	37,9	35,69	40,05
7	25,9	276,86	339,77	323,48	313,37
11	531	16335,43	19432,78	18260,85	18009,69
12	1044	37521,98	39854,86	42955,38	40110,74
13	2095	109042,77	143845,05	157771,45	136886,42
14	3128	202517,35	268550,01	270673,33	247246,9

Hasil pengujian waktu dari proses ekstraksi dapat dilihat pada Tabel 5.14. Proses ekstraksi membutuhkan waktu rata-rata hingga 4531,26 milisekon untuk berkas PDF *stego text* berukuran hingga 298 kB dan waktu hingga 12285,83 milisekon untuk mengekstraksi berkas PDF *stego text* berukuran . Dari hasil pengujian, meski tak selalu, pengunaan opsi genap sering kali memiliki waktu *running* tercepat dalam proses ekstraksi.

Tabel 5.14 Hasil Uji Coba Waktu Ekstraksi

Data	Ukuran *PDF Stego Text* (kB)	Waktu Rata-rata Ekstraksi (ms)			Waktu Rata-rata Total (ms)
		Opsi Baris Genap	Opsi Baris 1-5	Opsi Baris 1-4-7	
4	7,58	10	9,6	12,02	10,54
5	7,88	-	-	10,72	10,72
8	8,66	6,06	8,57	7,72	7,45
6	8,72	-	7,77	9,64	8,71
1	8,86	11,21	22,65	28,72	20,86
3	8,91	15,72	26,07	29,15	23,65
10	8,93	12,18	13,69	7,99	11,29
2	9,28	14,9	18,42	33,77	22,36
9	9,55	11,16	9,01	33,83	18
7	13,1	32,11	33,47	84,34	49,97
11	152	2036,74	1448,09	3769,54	2418,12
12	305	4241,59	2837,69	6514,49	4531,26
13	604	7768,35	6040,66	15041,88	9616,96
14	904	12140,06	8315,38	16402,06	12285,83

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Bab ini akan memaparkan kesimpulan yang diambil selama pengerjaan Tugas Akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

7.1. Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian sistem steganografi teks pada Aksara Sunda dengan pendekatan *line shift coding* dapat diambil kesimpulan sebagai berikut:

1. Penyisipan *secret message* pada dokumen teks Aksara Sunda dapat dilakukan dengan menyisipkan bit biner *secret message* ke dalam dokumen teks. Penyisipan bit biner dilakukan melalui pendekatan *line shift coding* yang mengubah posisi baris secara vertikal. Pergeseran posisi baris dapat dilakukan dengan menukar blok Unicode dari karakter Aksara Sunda yang terdapat pada baris yang digeser posisinya. Proses ekstraksi pada *stego text* dilakukan dengan mengecek blok Unicode dari baris-baris yang dianggap bergeser.
2. Tingkat kemiripan *covet text* dan *stego text* dari hasil pengujian MOS mendapat nilai 3, yang artinya *stego text* yang dihasilkan sistem cukup mirip dengan dokumen aslinya. Sementara jika dilihat dari tingkat kemiripan ukuran filenya, berkas PDF *stego text* memiliki ukuran yang lebih kecil dari *cover text* aslinya dengan rata-rata tingkat perbedaan mencapai 46,1 %.
3. *Secret message* yang diekstraksi dari *stego text* masih bisa didapatkan secara utuh. *Stego text* juga masih mampu bertahan dari proses serangan berupa *fotocopy* hingga 2 kali proses pencetakan ulang. Bit-bit biner hasil ekstraksi masih didapatkan kembali secara utuh.

4. Nilai *capacity ratio* rata-rata sistem sebesar 6,06 (bit/KB) dimana opsi geser baris 1-4-7 memiliki *capacity ratio* paling tinggi mencapai 6,56 (bit/KB). Sedangkan *capacity ratio* opsi geser baris 1-5 mencapai 6,50 (bit/KB) dan *capacity ratio* baris genap 5,11 (bit/KB). Penggunaan opsi geser baris 1-5 dibandingkan baris genap mampu meningkatkan daya tampung *cover text* dalam menyimpan bit biner *secret message* sebesar 1,39 (bit/KB). Sementara penggunaan opsi geser baris 1-4-7 akan meningkatkan *capacity ratio* hingga 1,45 (bit/KB) dibandingkan opsi geser baris genap.

7.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Diantaranya adalah sebagai berikut :

1. Modifikasi *line shift coding* dengan penggunaan metode lain bersamaan dengan *line shift coding* untuk meningkatkan *capacity ratio* sistem.
2. Penggunaan *library* yang mampu menangani *font* tipe OpenType Layout sehingga Aksara yang dihasilkan terlihat lebih sesuai.
3. Pengujian menggunakan aplikasi steganalisis untuk mengetahui apakah *stego text* yang dihasilkan terdeteksi menyembunyikan informasi rahasia.

DAFTAR PUSTAKA

[1] K. Sara, A. D. Mashallah and Y. M. Hossein, "A New Steganography Method Based On HIOP (Higher Intensity Of Pixel) Algorithm and Strassen's Matrix Multiplication," *Journal Of Global Research in Computer Science,* vol. 2, 2011.

[2] V. P.M. and V. Paul, "A Method for Text Steganography Using Malayalam Text," *Procedia Computer Science,* vol. 46, pp. 524-531, 2015.

[3] M. Nosrati, R. Karimi and M. Hariri, "An Introduction To Steganography Methods," *WAP Journal,* vol. 1, no. 3, pp. 191-195, 2011.

[4] R. S. R. Prasad and K. Alla, "A new approach to Telugu text steganography," in *Wireless Technology and Applications (ISWTA)*, Langkawi, 2011.

[5] M. Tezar, R. Magdalena and N. Andini, "Implementation and Analysis Of Message Security Using LSB Steganography and Relative Displacement Cipher Cryptography Algorithm," *e-Proceeding of Engineering,* vol. 2, no. 13, p. 7190, 2015.

[6] A. Pradhan, A. K. Sahu, G. Swain and K. R. Sekhar, "Performance Evaluation Parameters of Image," in *International Conference on Research Advances in Integrated Navigation Systems*, Bangalore, 2016.

[7] I. Andiniarti, "Implementasi Steganografi pada Media Teks dengan Metode Line-Shift Coding dan Metode Centroid," Central Library of Bogor Agricultural University, Bogor, 2009.

[8] B. Pfiztmann, "Information Hiding Terminology," in *Lecture Notes in Computer Science*, Cambridge, Springer, 1996, pp. 347-350.

[9] M. Agarwal, "Text Steganographic Approaches : A Comparison," *International Journal of Network Security & Its Applications (IJNSA)*, vol. V, no. 1, 2013.

[10] K. Bennett, "Linguistic Steganography : Survey, Analysis, and Robustness Concerns for Hiding Information in Text," Purdue University, West Lafayette, 2004.

[11] I. Baidillah, U. A. Darsa, O. Abdurahman, T. Permadi, G. Gunardi, A. Suherman, T. Ampera, H. S. Purba, D. T. Nugraha and D. Sutisna, Direktori Aksara Sunda Untuk Unicode, Bandung: Pemerintah Provinsi Jawa Barat, 2008.

[12] Kingsunda, "Panduan Lengkap Belajar Bahasa Sunda Mudah dan Menyenangkan," [Online]. Available: www.kingsunda.com/aksara-sunda-kaganga-rarangken/.

[13] "Java Logo PNG Transparent Background Download - DIY Logo Designs," diylogodesigns, [Online]. Available: www.diylogodesigns.com/blog/java-logo-png-transparent-background-download/.

[14] J. Gosling, B. Joy, S. G, G. Bracha and A. Buckley, The Java Language Specification, Seventh Edition, Redwood City, California: Oracle America, Inc., 2011.

[15] D. P. Nurjati, "Mengatasi Netbeans Tidak Bisa Dibuka - Menutup Sendiri," pramudito, [Online]. Available: www.pramudito.com/netbeans-tidak-bisa-dibuka.html.

[16] Sun Microsystems, "NetBeans IDE - Overview," Sun Microsystems, [Online]. Available: https://netbeans.org/features/index.html.

[17] M. Moore, "Apache PDFBox 2.0 is Released - SD Times," [Online]. Available: www.sdtimes.com/adobe/apache-pdfbox-2-0-is-released/.

[18] Apache, "Apache PDFBox," Apache, [Online]. Available: https://pdfbox.apache.org/.

[19] C. Water, "Apache POI list of Excel supported functions," Cedric Water, [Online]. Available:

https://waltercedric.com/index.php?option=com_content&view=article&id=2163&catid=102&Itemid=332.

[20] Tutorials Point, "Apache POI Tutorial," Tutorials Point, [Online]. Available: http://www.tutorialspoint.com/apache_poi/.

[21] GitHub, "GitHub - fontforge/fontforge," GitHub, [Online]. Available: https://github.com/fontforge/fontforge.

[22] FontForge, "FontForge Open Source Font Editor," FontForge, [Online]. Available: http://fontforge.github.io/en-US/.

[23] A. Suharto, "Puisi » Kumpulan Puisi-Puisi Terbaru 2017 - Atom," Loker Puisi, [Online]. Available: https://www.lokerpuisi.web.id/2012/07/puisibahasasunda.html.

[24] A. Karsana, "Manglé - Majalah Mingguan Basa Sunda | Apan Urang Lembur Tea," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1389097481.

[25] D. Héndrayana, "Manglé - Majalah Mingguan Basa Sunda | Dandanggula Sugan," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1421308303.

[26] A. Paskal, "Puisi » Kumpulan Puisi-Puisi Terbaru 2017 - Atom," Loker Puisi, [Online]. Available: https://www.lokerpuisi.web.id/2012/07/puisibahasasunda.html.

[27] A. Gunawan, "Manglé - Majalah Mingguan Basa Sunda | Kertas," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1389097414.

[28] A. Andriansyah, "Manglé - Majalah Mingguan Basa Sunda | Potret," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1396441109.

[29] G. Suwarna, "Sajak-Sajak Godi Suwarna | Salangit's Blog," Salangit's Blog, [Online]. Available:

https://salangit.wordpress.com/sastra/sajak-sajak-godi-suwarna/.

[30] Risnawati, "Manglé - Majalah Mingguan Basa Sunda | Sekar Merdika," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1428507075.

[31] A. Karsana, "Manglé - Majalah Mingguan Basa Sunda | Sinareng Bulan Purnama," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1396441029.

[32] C. R. Isnendes, "Manglé - Majalah Mingguan Basa Sunda | Tafakur Heula," Manglé, [Online]. Available: http://mangle-online.com/pidangan/perelean/1392880180.

LAMPIRAN

A. Data Pengujian MOS

Responden	Penilaian MOS					
	Skenario A	Skenario B	Skenario C	Skenario D	Skenario E	Skenario F
Responden 1	4	4	4	3	3	3
Responden 2	4	4	4	4	4	4
Responden 3	3	1	5	4	3	2
Responden 4	5	5	5	3	1	5
Responden 5	3	3	3	3	3	3
Responden 6	3	4	5	3	5	4
Responden 7	3	3	3	3	3	3
Responden 8	2	2	2	2	2	2
Responden 9	4	4	4	4	4	4
Responden 10	3	3	3	3	3	3
Responden 11	4	4	4	4	4	4
Responden 12	3	3	3	3	3	3
Responden 13	3	3	3	1	1	1
Responden 14	4	4	4	4	4	4
Responden 15	4	4	4	4	4	4
Responden 16	4	4	4	4	4	4
Responden 17	4	4	4	4	4	4
Responden 18	4	4	4	4	4	4
Responden 19	3	3	3	3	3	3
Responden 20	3	3	3	3	3	3
Responden 21	3	3	3	3	3	3
Responden 22	3	3	3	3	3	3
Responden 23	4	4	4	4	4	4
Responden 24	3	3	3	3	3	3
Responden 25	4	4	4	4	4	4
Responden 26	1	1	1	1	1	1
Responden 27	3	3	3	3	3	3
Responden 28	3	3	3	3	3	3
Responden 29	3	3	3	3	3	3
Responden 30	3	3	3	3	3	3
Responden 31	2	2	2	2	2	2
Responden 32	2	2	2	2	2	2
Responden 33	2	2	2	2	2	2
Responden 34	3	3	3	3	3	3
Responden 35	4	4	4	4	4	4
Nilai MOS	3,23	3,20	3,34	3,11	3,09	3,14

B. Isi Teks Anaking.docx

ᘜᒪᑎ...
ᖇᖱᑲᑫᖇᒪᓴ ᑭᖱᑘᖇᒥᒥᓴ ᘜᘐ ᔑᑲᑫᑫᓴ
ᕕᖇᒥᒥᓴ ᔑᖱᖇᑫᓴ ᖇᔑᔓ
ᒻ ᐃᖱᒥᒥ ᔓ ᐃᑘᓴᙆ
ᘜᒪᑎ..
ᔑᖱᖇ ᒻ ᒐᒐ ᑫᕕᑫᔑᑎᓴ ᘜᘐ
ᕕᐃᐃᐃᓴ ᐄᑭᖱ ᕕᒪᔑ ᒪᒪᓴᙆ ᒪᔓᒧ
ᘜᔑ ᑲᒪ ᑲᖇ ᘜᘐ
ᐃᐃᑫ ᐃᑘᓴᙆ ᖇᑫᙆ ᔑᑎ ᒻ ᘜᘐ
ᒪᙆᒪᓴ ᒥᒥᑲ ᕕᓴᒥᓴ ᑫᐁᑎ ᔓᖇᖇ
ᘜᘐ ᔑᔑᑫᓴ ᒪᑲᒥ ᒥ ᐃᑫᑫ
ᑫᖇᙆ ᒥᖇ ᖇᖇ ᒥᒥᔓᕕᓴᒪ
ᘜᒪᑎ..
ᔑᖱᑫᓴᑫᙆ ᕕᑲ ᒻ ᔕᐁ
ᑫᒪᓴ ᔑᑲᑫᓴ ᑫᑫᘪ ᑲᙆᘪᒪᓴ ᘜᘐ
ᐃᒥᒥᐁᓴ ᑘᕕᓴᔑ ᕕᐁᘪᕕᓴᒪ ᔕᐁ
ᘜᒪᑎ..
ᔑᑲᑫᓴ ᒉᕕᓴ ᒣᒪᓴᐁᒥᒥᓴ ᑫᕕᖇ ᑲᘪᕕ
ᑎᑫᔑᔑᓴ ᒪᐁᙆ ᒥᒥᘪᓴᘪ ᖇᔑ ᑲᘪᕕ
ᒉᖇ ᒥᑲᒻ ᑫᘪ ᐁᐁᒪ
ᐃᐃᑫ ᒻ ᔕᐁ ᔑᔑ ᐃᓴᘪ ᘪᕕᕕᓴ ᐃᓴᘪᓴᘪ ᑑᓴᔑᕕ
ᔑᔑ ᐃᓴᘪ ᔑᒪᙆ ᒉ ᐁᐃᘪ
ᒥᒥᓴᑑᓴ ᒻᒪ ᔕᐁ ᘪᒪᓴ ᐃᓴᘪ ᘪᓴ ‘ᘜ
ᕕᖇᒥᒥᓴ ᔑᑲᑫᓴ ᔑᖱᑫᓴ ᘪᑭᔓᙆ
ᒻᐃᐁᑎᔑᒥᓴ ᒥᒉ ᘪᐁᒪᓴ ᒻ ᔑᒥᒥᓴᘪ
ᐃᐁᑎᐁᙆᓴ ᘪᕕᒉ ᒉᐁᒪᓴᙆ ᔕᑎᑎ ᒻ ᕕᒉᘪ

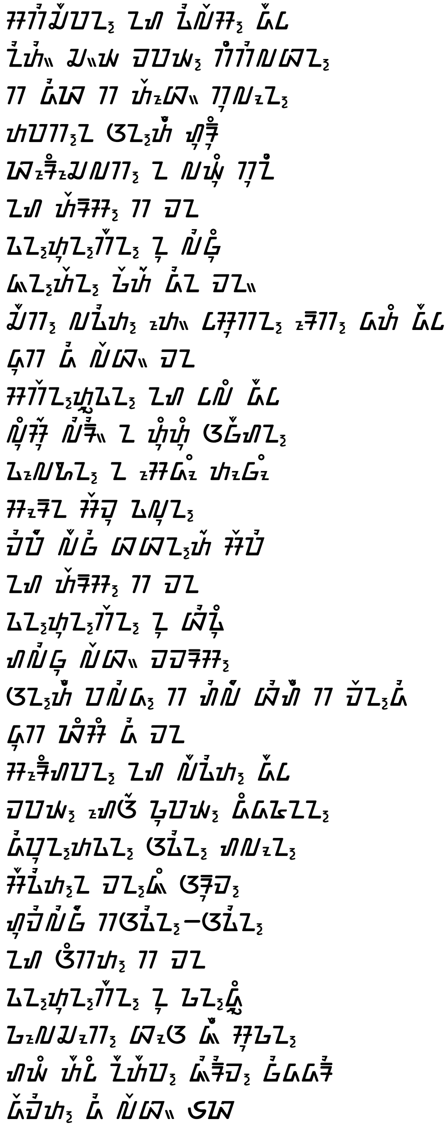
C. Isi Teks Apan Urang Lembur Tea.docx

[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]

D. Isi Teks Dandanggulan Sugan.docx

[illegible]

E. Isi Teks Hidup.docx



F. Isi Teks Katineung.docx

ꦱꦩꦏ ꦆꦲ꦳ ꦩꦶꦆꦭꦺꦆꦭ꧀
ꦥꦺꦤꦌꦭꦆꦭ꧀ ꦆꦲ꦳ ꦎꦎ ꦥꦲ꦳ꦽꦗꦆꦭ꧀ ꦲꦌꦮꦆꦭ꧀ ꦥꦽꦮꦆꦭ꧀
ꦌ꧀ ꦑꦲꦾ꧈, ꦲꦶꦭꦺꦩꦆ꧀, ꦲꦾ꧀ ꦗꦱꦴ
ꦤꦌꦤ꧀ ꦎꦌꦸ꧀–ꦎꦌꦸ꧀ ꦆꦺꦴꦎꦌ꧀
ꦒ꧀ꦏꦏꦶꦺꦆꦭ꧀ ꦮꦤꦆꦭ꧀
ꦱꦩꦏ ꦆꦲ꦳ ꦥꦾ꧈ꦏꦶꦺꦆꦭ꧀
ꦌ꧀ꦲꦩꦆ꧀ ꦤꦺꦭꦆ꧀ ꦆꦲ꦳ ꦎꦎ ꦌ꧀ꦌ꧀ꦲ꦳ꦩꦆꦭ꧀
ꦌ꧀ ꦍꦴꦊꦺꦍꦩ꧀ ꦩꦃ ꦑꦶꦆꦭ꧀ꦩꦃꦎꦎ꧀ ꦤꦺꦍꦩ꧀ꦆꦭ꧀
ꦣꦣꦮꦺꦤꦣꦆꦭ꧀–ꦣꦣꦮꦮꦺꦤꦣꦆꦭ꧀ ꦥꦌ꧀ꦒꦆ꧀ꦑꦲꦱꦆꦭ꧀
ꦱꦩꦏ ꦆꦲ꦳ ꦮꦩꦒꦆ꧀ꦶꦎꦆꦭ꧀
ꦩꦮꦌꦣꦃ ꦩꦃꦭ꧀ ꦎ ꦭꦺꦭ꧀ ꦭꦍ꧀ ꦥꦌ꧀ꦊꦌꦆꦭ꧀
ꦒꦩꦎꦆꦭ꧀ ꦤꦺꦍ꧀ ꦩꦃꦩꦃꦆꦮꦺꦆꦭ꧀
ꦎꦎꦺꦤ꧀ ꦎꦒꦺꦤꦺꦍ꧀ꦍ꧀ ꦍꦆꦭ꧀ ꦮꦶꦩ ꦌ꧀ ꦌꦮꦍ꧀
ꦎꦊꦆꦃꦣꦃ ꦥꦴꦣꦃ–ꦥꦴꦣꦃ ꦑꦶꦱꦆ꧀ ꦮꦒꦃꦥꦃ ꦆꦃꦒꦃꦣꦃ

G. Isi Teks Potret.docx

ĞŁĿ ŦŦĞŊŇ ĹĦŦĦĦŁ₂ĦĦŁ₂ ŇŇĻĦ ŊĠĦĦ₂
ŲĻŊĦ₂ ŇŇƆĦĦ₂ ŧŧŁ ĦŲ Ļ ĦŊĠĦĦ₂Ł ĻĻĦŁ₂ ĦĻƠĦ
ƆĦ ŲŇŁƆ ŁƠƆŁĦ ĦƠŁ₂ ĠĦŁ
ƆĦ ŇĬŧ ŁŊŇƆ₂ŁŴ ŊɀŊ₹ɀĻĻ ŧŧŧŧŇŇ₂
ŊĻ ĦĦĻ₂ ĠŁ₂Ĺ ŁŴĦĦĻŦŦŁ₂ ɀĴĠ ŧŦŦ₂Ĺ
ŁƠ ŦŦŇ ĦĦĬŴŧŧĦŁ₂ ŲɀɀĻĻŧŧ₂ ĦɀĹĦŧ₂
ĹŁ ĦĦŁĦ₂ ɀŧŧƆ₂ŁƠɀŦŦ₂ ŇĻŁ₂ĻĻŁ₂ ɀĦĦĦ ɀŧŁŦŦ₂
ĦĦĦ Ļ ŊŊ ŊĦ₂ŧŧĦ, ĴŊĦŦ₂ Ļ ƋĠŦŇŁ₂ ŦŦĦĦĴ
ŁĻŇ ŇŊŴ ŊŊƆ₂ŲŧŦŁ₂ ƆŁ₂ŧŧĦ ŁŴĦ ĹĬ ŇŇŧ
ƆŁ₂ŧŧĦ ɀĦĢ ĦĦĢ Ļ ŴŇĿ ĹĬĦĦ₂ŁƠ ĦĦ
ŲĻŊĦ₂ ĦŲ ŁŁŁ₂ ĹĹ Ļ ŦĻĢ ĹĬĦĿŴ₂ ŧŧŧŧŇŇ₂
ĦŁƠŁ₂ ŁĦŁ₂ ĹĬ ŧŧĻĦŇĦ Ļ ŦĻĦĦ ƋĦŁŁ₂
ŁĻŇ ŊŧŧŦ ĹĬĦĦƋ₂ ŁƋ₂ŁŴ Ļ ŊŁ ŦŦŁƠŁ₂ĹĬ ĦĦƋ
ŧŧĹĬ ŁƠɀĹĦĦ₂ ŊĿŧŧ₂ ŦŦƋ ŦŦŲŁ₂ ĦĦĹŴ₂Ł ŁƋ₂Ų̃ƆŁ₂
ĹŁ ŲĻĻĦ ĦŦŇ Ļ ŊĠĦĦ₂ ŇŲĿŧŧŁ₂ ŦŦŊŦŦ

H. Isi Teks Sajak-Sajak Godi Suwarna.docx

ꓲꓰꓱꓲ ꓱꓱꓵꓲ ꓵꓵꓲꓲ ꓷꓷꓲꓷ ꓲꓲ ꓵꓵꓲꓲ
ꓲꓲꓷꓲꓲ ꓷꓷꓷꓷꓲꓲ ꓲꓲꓷꓷꓲꓲ ꓲꓲ ꓷꓷꓲꓲ ꓲꓲ ꓷꓷꓲꓲ ꓲꓲ ꓷꓷꓲꓲ
ꓷꓲꓲꓲ ꓲꓲꓲꓲ
ꓷꓷꓵꓲꓲ ꓲꓲꓲꓲ ꓲꓲ ꓵꓵꓲꓲ ꓲꓲ ꓵꓵꓲꓲ ꓲꓲ ꓵꓵꓲꓲ
*ꓲꓲ ꓷꓷ

ᥣᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ
ᥣᥣᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ
ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ
ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ
ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ
*ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ ᥒ ᥣᥣᥲᥴᥣᥱᥲᥣ ᥐᥒᥲᥴᥣᥱᥲᥣ

I. Isi Teks Sekar Merdika.docx

ᱚᱵᱟᱱ ᱨᱮ ᱨᱮᱭᱦ ᱫᱩᱥᱤ,
ᱨᱮᱭᱦ ᱣᱟᱦᱤ ᱩᱱ ᱩᱱᱤᱭᱟᱜ ᱱᱟᱦ ᱩᱱᱤ,
ᱣᱭᱦᱷ ᱱᱟᱣ ᱱ ᱨᱮᱚᱱᱤᱫᱤ,
ᱣᱮᱮᱣ ᱨᱮᱣᱤᱨᱮ ᱦᱮᱮᱣᱩᱩᱦᱤ,
ᱱᱟᱤ ᱱᱩᱫᱤ ᱣᱮᱱᱟᱣᱤ ᱚᱮᱨᱮᱫᱤᱣᱤ,
ᱨᱮᱚᱮᱣᱟᱱᱣᱤ ᱣᱮᱪ ᱫᱩᱥᱤ,
ᱡᱤᱪ ᱩᱱᱤᱭᱦ ᱩᱦᱦ ᱤᱪ.
ᱫᱩᱥᱤ ᱢᱮᱦ ᱨᱱ ᱩᱩᱣᱩ,
ᱩᱣᱮᱣ ᱦᱦᱤᱭ ᱩᱚᱚᱤ ᱩᱵ ᱣᱤᱤᱪ ᱡᱤᱪ,
ᱚᱵᱩᱵᱣᱣᱤ ᱚᱚᱤᱦ ᱣᱤᱤᱪ,
ᱤᱪ ᱦᱦᱣᱦ ᱚᱩᱣᱦᱪᱤᱣᱤ,
ᱚᱤᱦ ᱦᱦᱣᱦ ᱩᱱᱮᱱᱮ ᱦᱦᱚᱱᱤ ᱦᱦ ᱚᱣᱮᱦᱤ,
ᱚᱦᱦᱱᱤ ᱩᱱᱦᱦ ᱩ ᱩᱡᱦᱣᱣᱤ,
ᱚᱮᱩᱚᱚᱤ ᱩᱮᱦᱦᱣᱤ ᱫᱫᱚᱤᱚᱱ ᱤᱪ.
ᱫᱦᱱᱱ ᱦᱦᱣᱤᱩᱱᱤ
ᱫᱫᱦ ᱩᱩ ᱚᱚᱣᱫᱫᱣᱤ
ᱫᱫᱱᱱᱣᱤ–ᱫᱫᱱᱱᱣᱤ ᱩᱡᱦᱣᱤᱣᱤ ᱱᱱ ᱱᱮᱦᱦ ᱫᱩᱥᱤ
ᱤᱫᱫᱦᱦᱣ ᱩᱚᱚᱤ ᱩᱣᱤᱩᱦᱦ
ᱦᱱᱦᱫ ᱱᱦᱦᱱᱤ ᱩᱦᱦ ᱩ ᱩᱡᱦᱣᱤᱣᱤ
ᱡᱤᱩᱦᱱ ᱩ ᱱᱱᱱᱮᱣᱤᱡᱫ
ᱱᱦᱩᱮᱣᱤ ᱥᱣᱤᱮᱦᱦ ᱣᱚᱤᱡᱱᱱᱤ ᱢᱦᱣ
ᱩᱣᱦᱫ ᱱᱱᱦᱦ ᱱᱚᱮᱚᱣ
ᱡᱤᱦᱡᱫ ᱤᱪ ᱡᱫᱩᱣᱤ ᱮᱩᱣᱤᱦᱱ

J. Isi Teks Sinareng Bulan Purnama.docx

[illegible]

K. Isi Solawat Daun.docx

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Muhsin Bayu Aji Fadhillah, lahir pada tanggal 16 November 1996 di Sukoharjo, Jawa Tengah. Penulis menempuh pendidikan mulai dari SD Negeri Gayam 1 (2003-2009), SMP Negeri 1 Sukoharjo (2009 – 2012) dan SMA Negeri 1 Sukoharjo (2012-2014). Saat ini penulis sedang menempuh pendidikan perguruan tinggi di Institut Teknologi Sepuluh Nopember Surabaya di jurusan Informatika Fakultas Teknologi Informasi dan Komputasi angkatan tahun 2014.

Komunikasi dengan penulis dengan senang hati dilayani dan dapat melalui email langsung ke : muhsin21baf@gmail.com atau muhsin14@mhs.if.its.ac.id .

[Halaman ini sengaja dikosongkan]